Aalto University
School of Science
and Technology

Sharing Resources Between AES and the SHA-3 Second Round Candidates Fugue and Grøstl

Kimmo Järvinen

Department of Information and Computer Science
Aalto University, School of Science and Technology
Espoo, Finland

AES-inspired SHA-3 Candidates

- Design strongly influenced by AES:
  Share the structure and have significant similarity in transformations, or even use AES as a subroutine
- ECHO, Fugue, Grøstl, and SHAvite-3
- Benadjila et al. (ASIACRYPT 2009) studied useability of Intel's AES instructions for AES-inspired candidates
  Conclusion: only ECHO and SHAvite-3, which use AES as a subroutine, benefit from the instructions
- This is the first study of combining AES with the SHA-3 candidates on hardware (FPGA)

Research Topics and Motivation

Research Questions

What modifications are required to embed AES into the data path of the hash algorithm (or vice versa)?

How much resources can be shared (logic, registers, memory, . . .)?

What are the costs (area, delay, throughput, power consumption, . . .)?

Applications

Any applications that require dedicated hardware implementations of a hash algorithm and a block cipher would benefit from reduced costs

Particularly important if resources are very limited

Advanced Encryption Standard

AES with a 128-bit key (AES-128)

State: 4×4 bytes; each byte is an element of $GF(2^8)$

10 rounds with four transformations

Transformations

SubBytes: Bytes mapped independently with (1) a multiplicative inverse in $GF(2^8)$ and (2) an affine transformation

ShiftRows: The row i shifted to the left by i bytes

MixColumns: Columns multiplied with a fixed polynomial over $GF(2^8)$ modulo $x^4 + 1$ (omitted in the last round)

AddRoundKey: A 128-bit bitwise xor with a round key

Fugue

32-bit block size, 960-bit chaining value

AES-inspired SMIX and certain other transformations (xors and rotations)

SMIX operates on 128 bits

SMIX includes SubBytes of AES followed by Super-Mix inspired by MixColumns

Super-Mix includes cross-mixing between columns and can be seen as a matrix multiplication where a 16-byte vector is multiplied from the left by a 16×16 byte matrix (sparse)

Fugue / AES

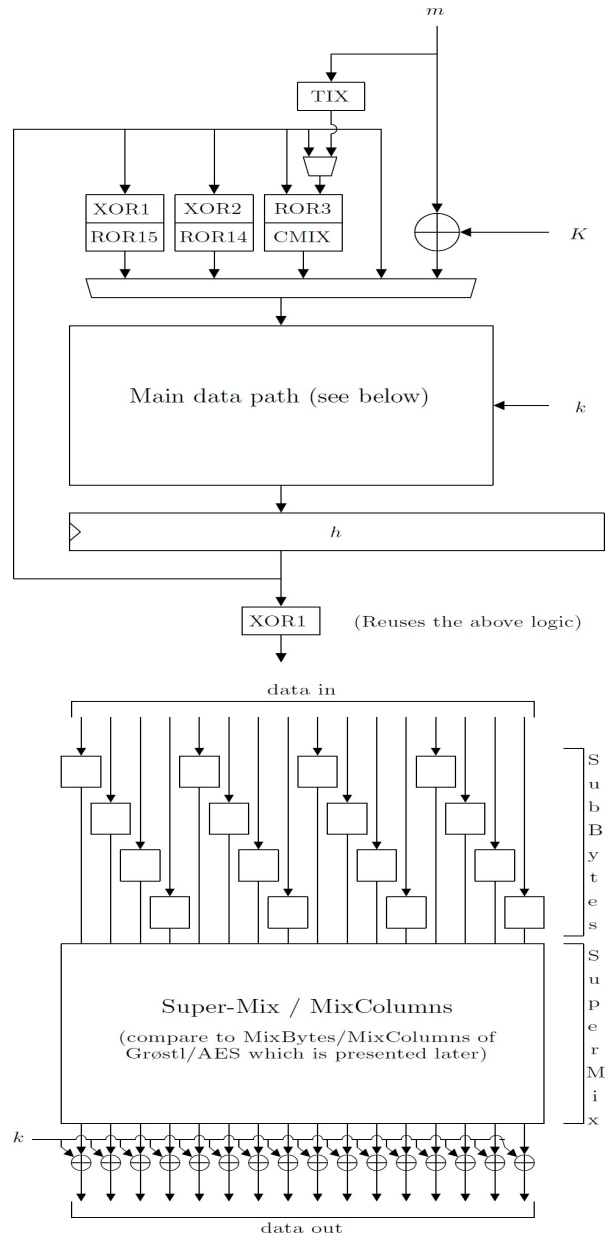
SubBytes shared entirely

ShiftRows embedded into the input multiplexers

Super-Mix/MixColumns share the multipliers and xors but require additional multiplexers

Interface mismatch (inputs 32 vs. 128 bits, outputs 256 vs. 128 bits)

KeyExpansion on the data path can share four S-boxes and reuse registers (*h*) but doubles the latency



Grøstl

512-bit block size, 512-bit chaining value

The compression function consists of two AES-inspired transformations: P and Q which are almost the same

P and Q include AddRoundConstant (the only difference between P and Q), SubBytes, ShiftBytes, and MixBytes

The transformations are applied to a 512-bit State

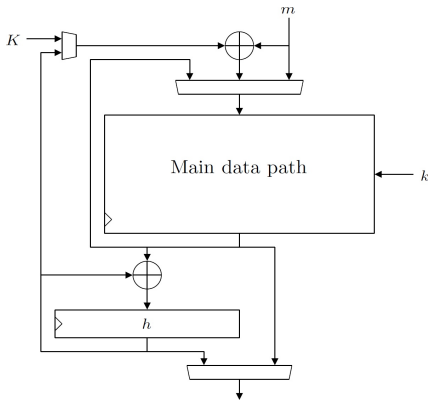
Grøstl / AES

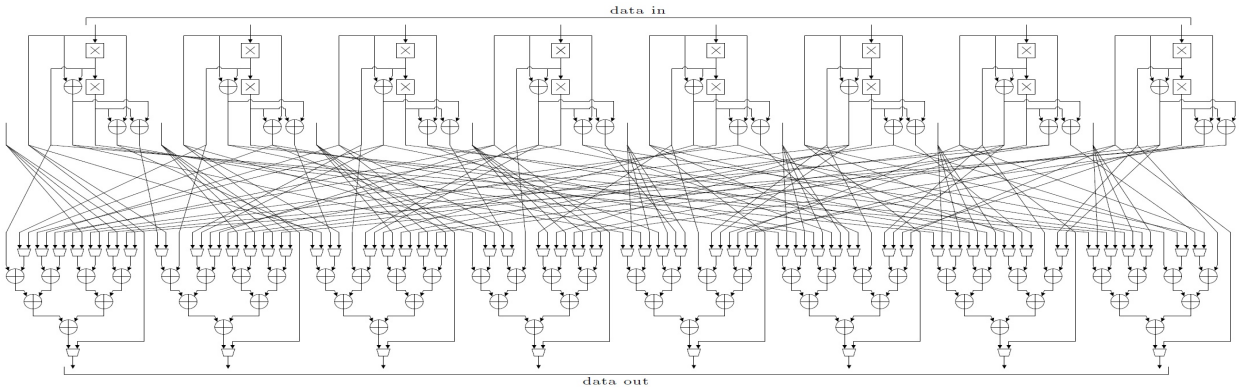
The 512-bit Grøstl data path used for at most four parallel AES encryptions (or KeyExpansions)

SubBytes shared entirely

ShiftRows constructed from ShiftBytes by swapping some bytes (12 if four parallel AES encryptions)

MixBytes/MixColumns share the multipliers and xors but require additional multiplexers





Results

Table: Fugue results on Altera Cyclone III EP3C80F780C7 FPGA

	`fugue`	`fugue_aes`	`fugue_aes_ke`
Place&route results			
Logic cells (LC)	3562	4520 (+26.9 %)	4875 (+36.9 %)
Registers	1005	1105 (+10.0 %)	1113 (+10.7 %)
f_{max} (MHz)	63.93	60.75 (−5.0 %)	59.81 (−6.4 %)
Fugue performance			
Latency (clock cyc.)	2	2	2
Throughput (Gbps)	1.023	0.972	0.957
AES performance			
Latency (clock cyc.)	–	10	20
Throughput (Gbps)	–	0.778	0.383

Note: AES core with KeyExpansion requires 2525 LCs and 527 registers (of which KeyExpansion takes 536 LCs and 136 regs.)

Results (cont.)

Table: Grøstl results on Altera Cyclone III EP3C80F780C7 FPGA

	groestl	groestl_aes	groestl_aes_ke	groestl_4aes
Place&route results				
Logic cells (LC)	12086	12387 (+2.5%)	12520 (+3.6%)	13723 (+13.5%)
Registers	1547	1550 (+0.2%)	1558 (+0.7%)	1550 (+0.2%)
f_{max} (MHz)	57.52	54.13 (−5.9%)	55.79 (−3.0%)	56.03 (−2.6%)
Grøstl performance				
Latency (clock cyc.)	20	20	20	20
Throughput (Gbps)	1.473	1.386	1.428	1.434
AES performance				
Latency (clock cyc.)	–	10	10	10
Throughput (Gbps)	–	0.693	0.714	2.869

Conclusions

Both Fugue and Grøstl can be combined with AES with small overheads in area and speed (at least in FPGAs)

Grøstl has almost negligible overheads because the entire data path and registers can be shared in a direct manner and including parallel encryptions and KeyExpansion(s) is easy

Possibility to efficiently combine the hash algorithm with AES is an asset that should be taken into account while selecting SHA-3

Future work: Other data path widths, unrolling and pipelining, different algorithm variants, side-channel countermeasures, . . .

Thank you.
Questions?