Compact FPGA Implementations of Selected Round 3 SHA-3 Candidates

Bernhard Jungk
bernhard.jungk@hs-rm.de

Hochschule RheinMain
Wiesbaden, Germany

CryptArchi 2011
Ruhr-Universität Bochum

Compact Implemtations

Already done:

- Skein, JH and Grøstl
- Evaluation currently for Virtex-5 FPGAs only

Near future:

- Implement BLAKE and Keccak
- Improve benchmarking
  - More platforms (Altera, Spartan-3, Spartan-6, Virtex-6, ...)
  - Usage of a modified XBX framework for automated benchmarking and power measurements
  - Report performance for short messages

Compact Implemtations

Already done:

- Skein, JH and Grøstl
- Evaluation currently for Virtex-5 FPGAs only

Near future:

- Implement BLAKE and Keccak
- Improve benchmarking
  - More platforms (Altera, Spartan-3, Spartan-6, Virtex-6, ...)
  - Usage of a modified XBX framework for automated benchmarking and power measurements
  - Report performance for short messages

Compact Implemtations

Already done:

- Skein, JH and Grøstl
- Evaluation currently for Virtex-5 FPGAs only

Near future:

- Implement BLAKE and Keccak
- Improve benchmarking
  - More platforms (Altera, Spartan-3, Spartan-6, Virtex-6, ...)
  - Usage of a modified XBX framework for automated benchmarking and power measurements
  - Report performance for short messages

Compact Implemtations

Already done:

- Skein, JH and Grøstl
- Evaluation currently for Virtex-5 FPGAs only

Near future:

- Implement BLAKE and Keccak
- Improve benchmarking
  - More platforms (Altera, Spartan-3, Spartan-6, Virtex-6, ...)
  - Usage of a modified XBX framework for automated benchmarking and power measurements
  - Report performance for short messages

Compact Implemtations

Already done:

- Skein, JH and Grøstl
- Evaluation currently for Virtex-5 FPGAs only

Near future:

- Implement BLAKE and Keccak
- Improve benchmarking
  - More platforms (Altera, Spartan-3, Spartan-6, Virtex-6, ...)
  - Usage of a modified XBX framework for automated benchmarking and power measurements
  - Report performance for short messages

Compact Implemtations

Already done:

- Skein, JH and Grøstl
- Evaluation currently for Virtex-5 FPGAs only

Near future:

- Implement BLAKE and Keccak
- Improve benchmarking
    - More platforms (Altera, Spartan-3, Spartan-6, Virtex-6, ...)
    - Usage of a modified XBX framework for automated benchmarking and power measurements
    - Report performance for short messages

Compact Implemtations

Already done:

- Skein, JH and Grøstl
- Evaluation currently for Virtex-5 FPGAs only

Near future:

- Implement BLAKE and Keccak
- Improve benchmarking
  - More platforms (Altera, Spartan-3, Spartan-6, Virtex-6, ...)
  - Usage of a modified XBX framework for automated benchmarking and power measurements
  - Report performance for short messages

Design Constraints

- Overall goal: Minimal area
- No usage of BlockRAMs, DSP units
- Include the padding function
- Identical hardware interface

Design Constraints

- Overall goal: Minimal area
- No usage of BlockRAMs, DSP units
- Include the padding function
- Identical hardware interface

Design Constraints

- Overall goal: Minimal area
- No usage of BlockRAMs, DSP units
- Include the padding function
- Identical hardware interface

Design Constraints

- Overall goal: Minimal area
- No usage of BlockRAMs, DSP units
- Include the padding function
- Identical hardware interface

Hardware Interface

- Fast Simplex Link
- 32 bit wide unidirectional link with handshaking and an optional FIFO buffer

- 32 bit data signal
- *Master* FIFO full
- *Master* write
- *Slave* data available signal
- *Slave* read

Hardware Interface

- Fast Simplex Link
- 32 bit wide unidirectional link with handshaking and an optional FIFO buffer

- 32 bit data signal
- *Master* FIFO full
- *Master* write
- *Slave* data available signal
- *Slave* read

Hardware Interface

- Fast Simplex Link
- 32 bit wide unidirectional link with handshaking and an optional FIFO buffer

- 32 bit data signal
- *Master* FIFO full
- *Master* write
- *Slave* data available signal
- *Slave* read

Hardware Interface

- Fast Simplex Link
- 32 bit wide unidirectional link with handshaking and an optional FIFO buffer

- 32 bit data signal
- *Master* FIFO full
- *Master* write
- *Slave* data available signal
- *Slave* read

Hardware Interface

- Fast Simplex Link
- 32 bit wide unidirectional link with handshaking and an optional FIFO buffer

- 32 bit data signal
- *Master* FIFO full
- *Master* write
- *Slave* data available signal
- *Slave* read

Hardware Interface

- Fast Simplex Link
- 32 bit wide unidirectional link with handshaking and an optional FIFO buffer

- 32 bit data signal
- *Master* FIFO full
- *Master* write
- *Slave* data available signal
- *Slave* read

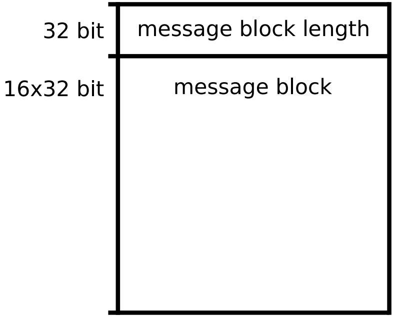
Hardware Interface

- Fast Simplex Link
- 32 bit wide unidirectional link with handshaking and an optional FIFO buffer

- 32 bit data signal
- *Master* FIFO full
- *Master* write
- *Slave* data available signal
- *Slave* read

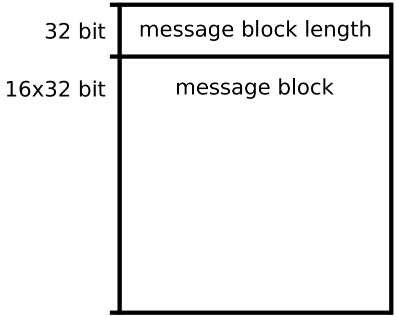
Hardware Interface

- Data format
- Message block length encoded in 32 bits
- Message block encoded with big endianess (512 bit=16x32 bit)



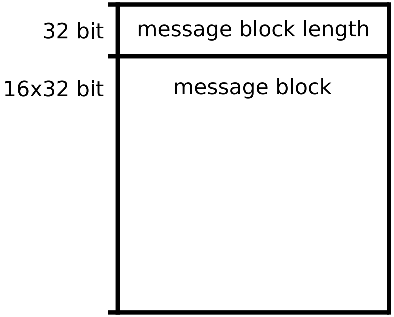
Hardware Interface

- Data format
- Message block length encoded in 32 bits
- Message block encoded with big endianess (512 bit=16x32 bit)

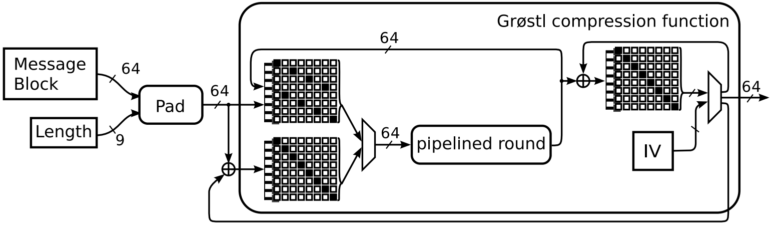


Hardware Interface

- Data format
- Message block length encoded in 32 bits
- Message block encoded with big endianess (512 bit=16x32 bit)

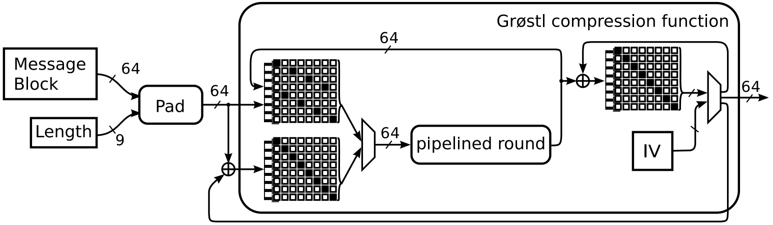


Grøstl - Design Overview



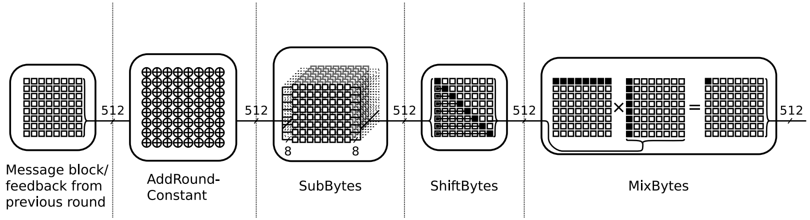
- Hardware for P and Q shared
- Distributed RAMs for P, Q and h

Grøstl - Design Overview



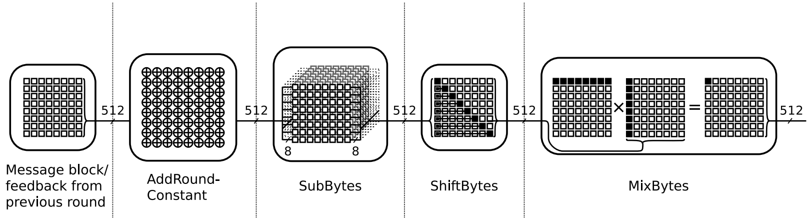
- Hardware for P and Q shared
- Distributed RAMs for P, Q and h

Grøstl - Round Function



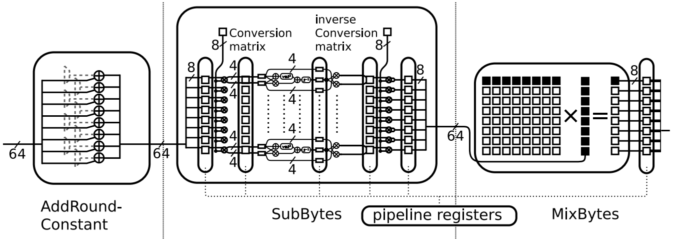
- 10 rounds
- Duplicated permutation (P and Q)

Grøstl - Round Function



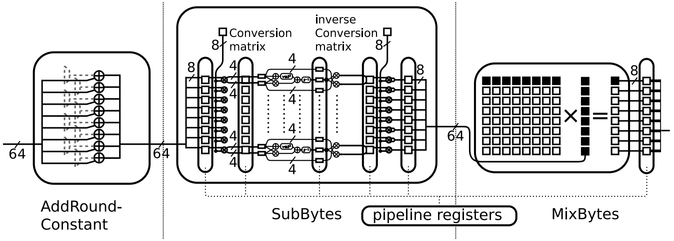
- 10 rounds
- Duplicated permutation (P and Q)

Grøstl - Compact Round Function



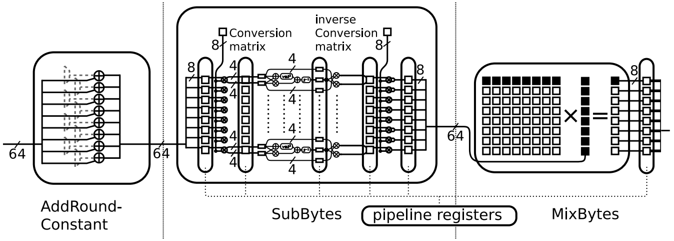
- Reduced datapath (64 bit)
- SubBytes implementation with composite field arithmetic

Grøstl - Compact Round Function



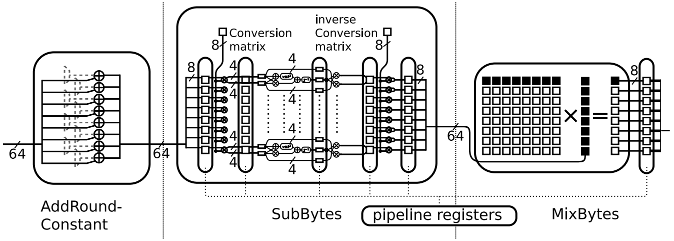
- Reduced datapath (64 bit)
- SubBytes implementation with composite field arithmetic

Grøstl - Compact Round Function



- Many clock cycles necessary (factor 8 increase)
- Additional pipeline registers increase the max. clock frequency

Grøstl - Compact Round Function



- Many clock cycles necessary (factor 8 increase)
- Additional pipeline registers increase the max. clock frequency

Grøstl - Number of clock cycles

- 10 rounds ($\times 10$)
- P/Q permutation ($\times 2$)
- Folding ($\times 8$)
- Overall: **160** clock cycles

Grøstl - Number of clock cycles

- 10 rounds ($\times$10)
- P/Q permutation ($\times$2)
- Folding ($\times$8)
- Overall: **160** clock cycles

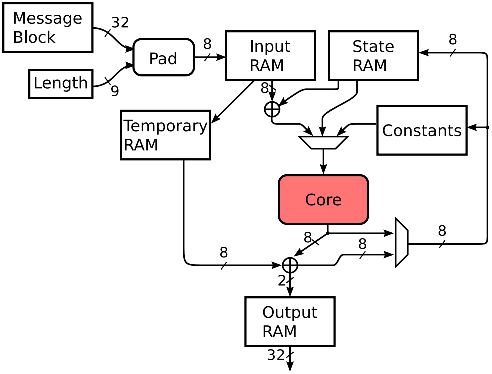
Grøstl - Number of clock cycles

- 10 rounds ($\times 10$)
- P/Q permutation ($\times 2$)
- Folding ($\times 8$)
- Overall: **160** clock cycles

Grøstl - Number of clock cycles

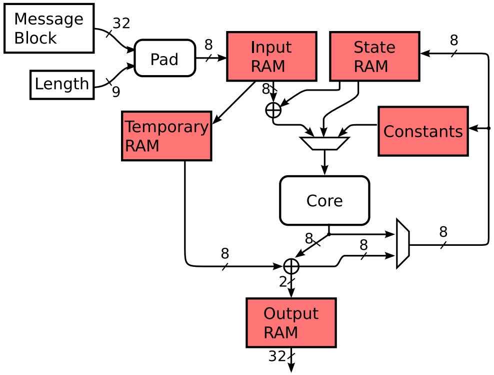
- 10 rounds ($\times 10$)
- P/Q permutation ($\times 2$)
- Folding ($\times 8$)
- Overall: **160** clock cycles

JH - Design Overview



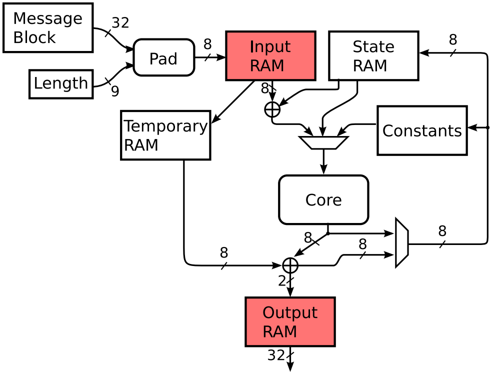
- Shared core for constants computation and round function

JH - Design Overview



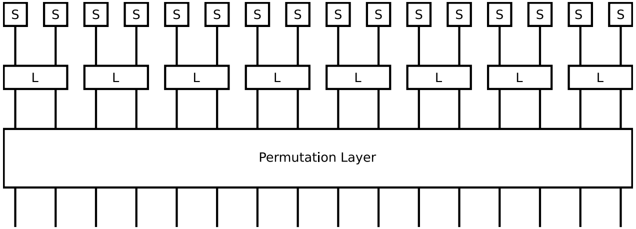
- RAMs for input, output, state, constants and a temp. buffer

JH - Design Overview



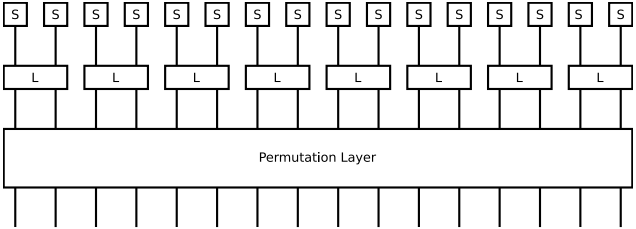
- Input and output RAMs do the grouping/de-grouping

JH - Round Function



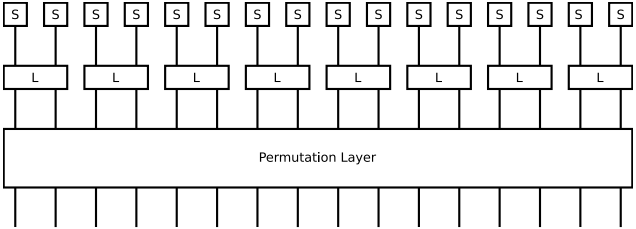
- S-Box, linear transformation and permutation
- Simpler transformations than Grøstl and thus faster per round
- But ... 42 rounds

JH - Round Function



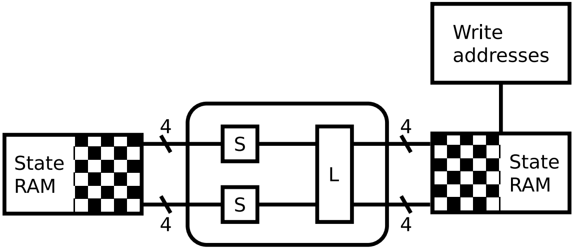
- S-Box, linear transformation and permutation
- Simpler transformations than Grøstl and thus faster per round
- But ... 42 rounds

JH - Round Function



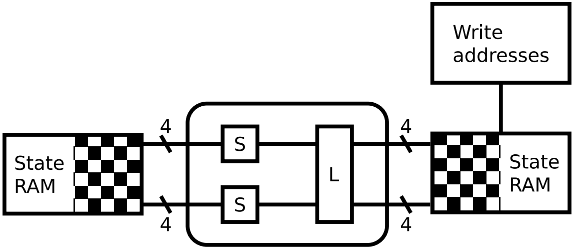
- S-Box, linear transformation and permutation
- Simpler transformations than Grøstl and thus faster per round
- But ... 42 rounds

JH - Compact Core



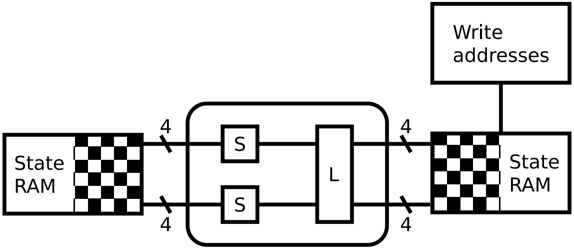
- 8 bit datapath
- Core consisting of S-Box and linear transformation
- Permutation is achieved by calculating write addresses to the internal RAM

JH - Compact Core



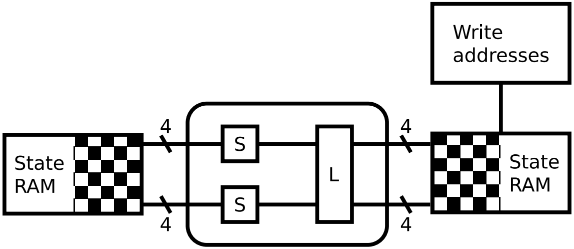
- 8 bit datapath
- Core consisting of S-Box and linear transformation
- Permutation is achieved by calculating write addresses to the internal RAM

JH - Compact Core



- 8 bit datapath
- Core consisting of S-Box and linear transformation
- Permutation is achieved by calculating write addresses to the internal RAM

JH - Compact Core



- 8 bit datapath
- Core consisting of S-Box and linear transformation
- Permutation is achieved by calculating write addresses to the internal RAM

JH - Number of clock cycles

- 42 rounds ($\times$42)
- Folding ($\times$160)
- Overall: **6720** clock cycles

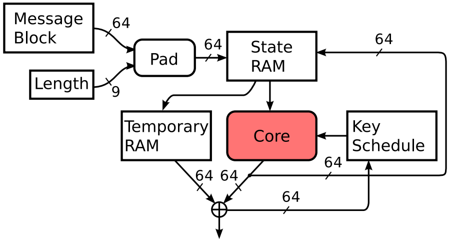
JH - Number of clock cycles

- 42 rounds (×42)
- Folding (×160)
- Overall: **6720** clock cycles

JH - Number of clock cycles

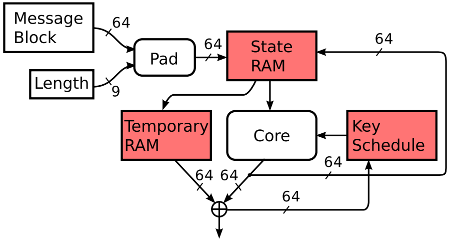
- 42 rounds ($\times 42$)
- Folding ($\times 160$)
- Overall: **6720** clock cycles

Skein - Design Overview



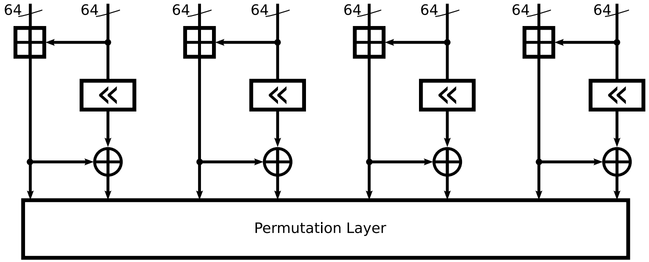
- Minimized, pipelined core (partial round function)

Skein - Design Overview



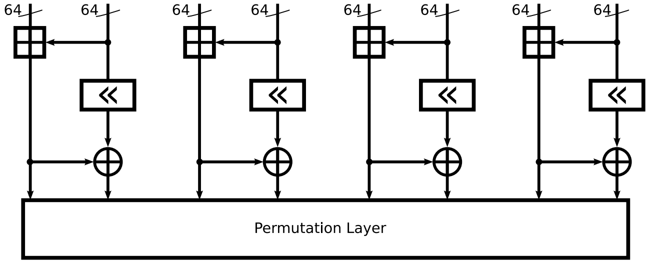
- RAMs for hash state, key schedule and a temporary buffer.

Skein - Round Function



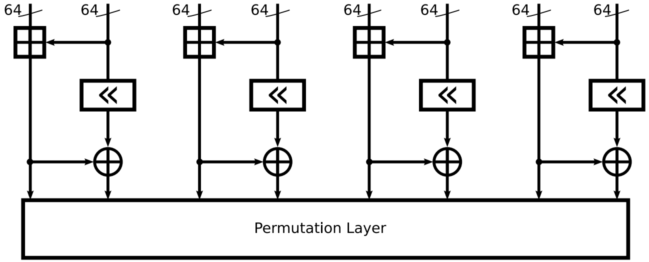
- 72 rounds
- Key injection (addition in 64 bit blocks) every 4th round

Skein - Round Function



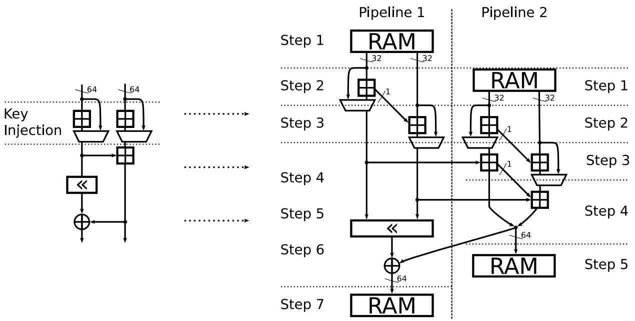
- 72 rounds
- Key injection (addition in 64 bit blocks) every 4th round

Skein - Round Function



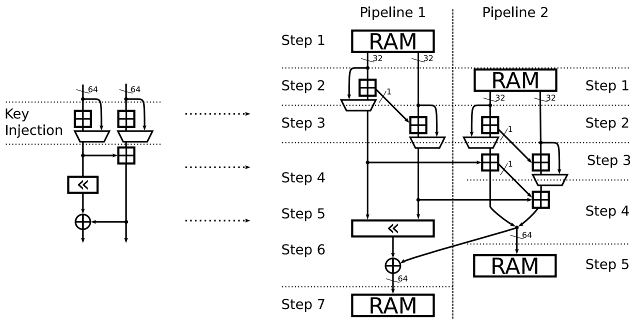
- 72 rounds
- Key injection (addition in 64 bit blocks) every 4th round

Skein - Compact Implementation



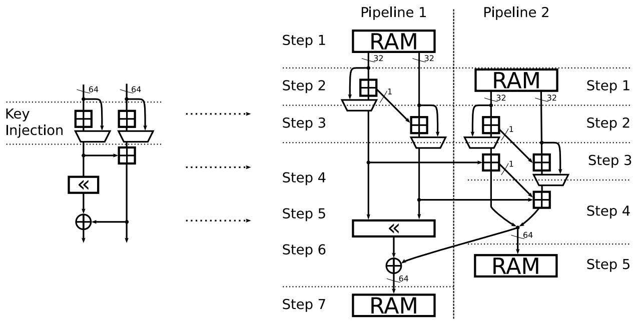
- 64 bit datapath
- Only 3 32 bit adder
- Non-optimal pipeline depth (caused by the permutation)

Skein - Compact Implementation



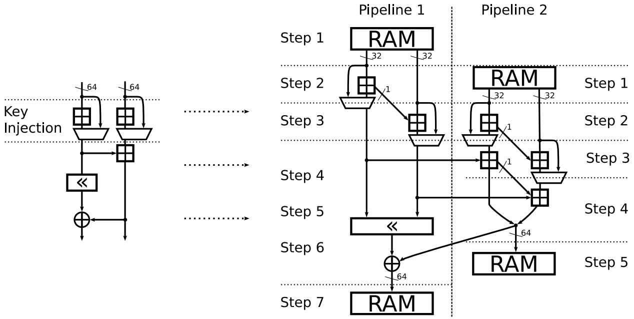
- 64 bit datapath
- Only 3 32 bit adder
- Non-optimal pipeline depth (caused by the permutation)

Skein - Compact Implementation



- 64 bit datapath
- Only 3 32 bit adder
- Non-optimal pipeline depth (caused by the permutation)

Skein - Compact Implementation



- 64 bit datapath
- Only 3 32 bit adder
- Non-optimal pipeline depth (caused by the permutation)

Skein - Number of clock cycles

- 72 rounds ($\times 72$)
- Folding ($\times 8$)
- Overall: **336** clock cycles

Skein - Number of clock cycles

- 72 rounds ($\times 72$)
- Folding ($\times 8$)
- Overall: **336** clock cycles

Skein - Number of clock cycles

- 72 rounds ($\times$72)
- Folding ($\times$8)
- Overall: **336** clock cycles

Results

Algorithm	Slices	MHz	MBit/s	$\frac{\text{MBit/s}}{\text{Slice}}$
Grøstl	470	354	**1132**	**2.40**
JH	**205**	341	27	0.13
Skein	555	271	237	0.42

Table: Results for Virtex-5 FPGAs

- Throughput for very long messages
- Xilinx ISE 12.3
- Optimized parameters for timing performance and area (xst, map, par)

Results

Algorithm	Slices	MHz	MBit/s	$\frac{\text{MBit/s}}{\text{Slice}}$
Grøstl	470	354	**1132**	**2.40**
JH	**205**	341	27	0.13
Skein	555	271	237	0.42

Table: Results for Virtex-5 FPGAs

- Throughput for very long messages
- Xilinx ISE 12.3
- Optimized parameters for timing performance and area (xst, map, par)

Results

Algorithm	**Slices**	**MHz**	**MBit/s**	$\frac{\textbf{MBit/s}}{\textbf{Slice}}$
Grøstl	470	354	**1132**	**2.40**
JH	**205**	341	27	0.13
Skein	555	271	237	0.42

Table: Results for Virtex-5 FPGAs

- Throughput for very long messages
- Xilinx ISE 12.3
- Optimized parameters for timing performance and area (xst, map, par)

Conclusions?

- Grøstl seems to be very flexible:
  - Smaller implementation possible (e.g. usage of only one S-box)
- JH is flexible, too, but:
  - The 8 bit data path was a bad choice
- Skein is harder to get small with a reasonable troughput:
  - The wide adder is one of the main problems
  - The rotation is making it hard for the router
  - The permutation prevents a pipeline with (optimal) depth 8

Conclusions?

- Grøstl seems to be very flexible:
  - Smaller implementation possible (e.g. usage of only one S-box)
- JH is flexible, too, but:
  - The 8 bit data path was a bad choice
- Skein is harder to get small with a reasonable troughput:
  - The wide adder is one of the main problems
  - The rotation is making it hard for the router
  - The permutation prevents a pipeline with (optimal) depth 8

Conclusions?

- Grøstl seems to be very flexible:
  - Smaller implementation possible (e.g. usage of only one S-box)
- JH is flexible, too, but:
  - The 8 bit data path was a bad choice
- Skein is harder to get small with a reasonable troughput:
  - The wide adder is one of the main problems
  - The rotation is making it hard for the router
  - The permutation prevents a pipeline with (optimal) depth 8

Questions?