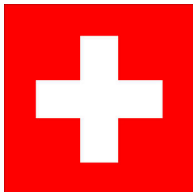


BLAKE



SWISS MADE

Jean-Philippe Aumasson

FHNW Windisch, Switzerland

Luca Henzen

ETH Zürich, Switzerland

Willi Meier

FHNW Windisch, Switzerland

Raphael Phan

Loughborough Uni, UK

BLAKE-64

for 64-bit words and 64-byte digests
10 cycles/byte



BLAKE-32

for 32-bit words and 32-byte digests
16 cycles/byte

the ChaCha function

bijective nonlinear transform of 4 words

$$a \oplus= b \qquad d = (a \oplus d) \lll 16$$

$$c \oplus= d \qquad b = (b \oplus c) \lll 12$$

$$a \oplus= b \qquad d = (a \oplus d) \lll 8$$

$$c \oplus= d \qquad b = (b \oplus c) \lll 7$$

BLAKE based on tweaked ChaCha

repeated 80 times in BLAKE-32

$a += m_j \oplus \text{const}_j$

$a += b$

$d = (a \oplus d) \ggg 16$

$c += d$

$b = (b \oplus c) \ggg 12$

$a += m_j \oplus \text{const}_j$

$a += b$

$d = (a \oplus d) \ggg 8$

$c += d$

$b = (b \oplus c) \ggg 7$

BLAKE based on tweaked ChaCha

repeated 112 times in BLAKE-64

$a += m_j \oplus \text{const}_j$

$a += b$

$c += d$

$a += m_j \oplus \text{const}_j$

$a += b$

$c += d$

$d = (a \oplus d) \ggg 32$

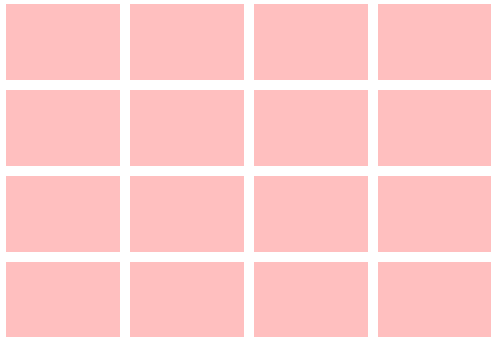
$b = (b \oplus c) \ggg 25$

$d = (a \oplus d) \ggg 16$

$b = (b \oplus c) \ggg 11$

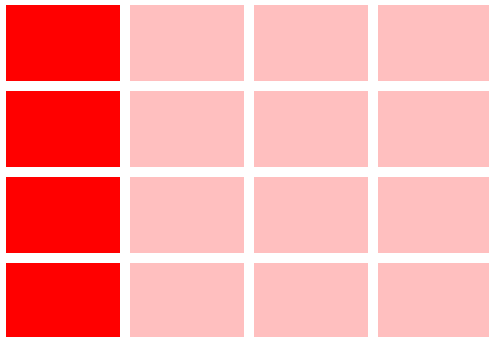
compression function state

initialized with salt, counter, chaining value



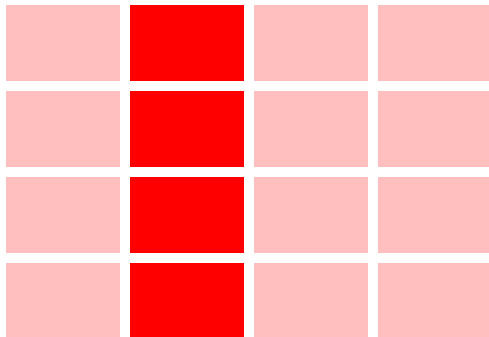
BLAKE round

apply the ChaCha function to each column



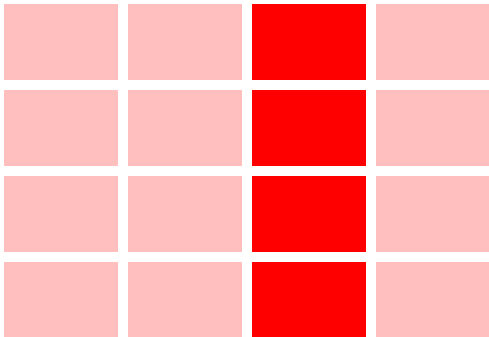
BLAKE round

apply the ChaCha function to each column



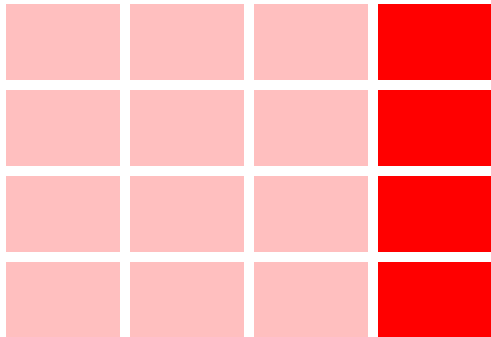
BLAKE round

apply the ChaCha function to each column



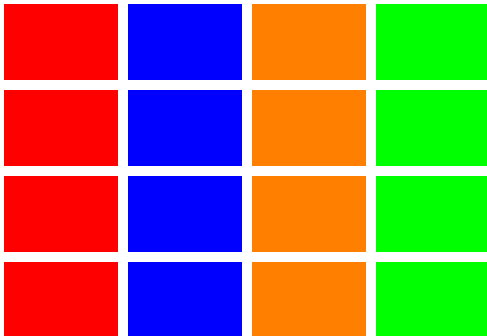
BLAKE round

apply the ChaCha function to each column



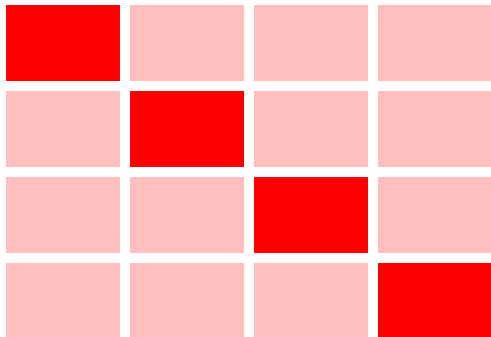
BLAKE round

or to all columns in parallel



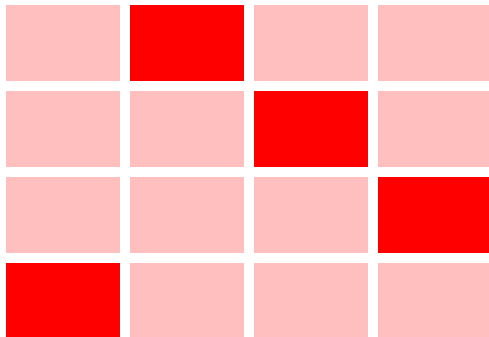
BLAKE round

apply the ChaCha function to each diagonal



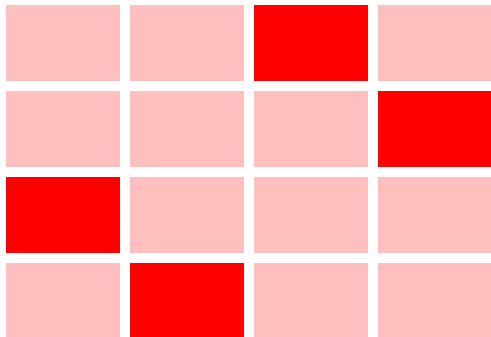
BLAKE round

apply the ChaCha function to each diagonal



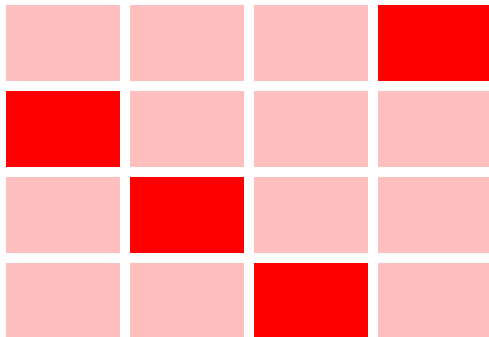
BLAKE round

apply the ChaCha function to each diagonal



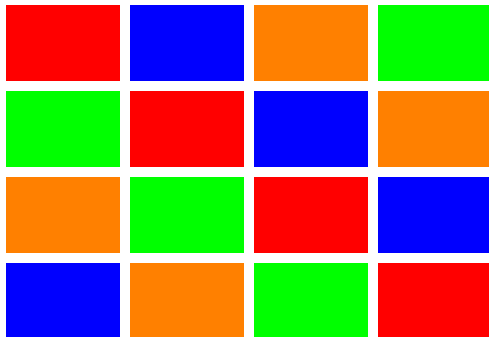
BLAKE round

apply the ChaCha function to each diagonal



BLAKE round

or to all diagonals in parallel



14 rounds for BLAKE-64

10 rounds for BLAKE-32

full diffusion in 2 rounds

best attack on ChaCha on 3.5 rounds

best attack on BLAKE on 2 rounds

BLAKE's iteration mode

(simplified HAIFA)

randomized hashing

RO-indifferentiability

2^n second preimage resistance

preserves collision and preimage resistance

BLAKE in hardware

straightforward implementation

various trade-offs speed/area

small memory requirements

our implementations

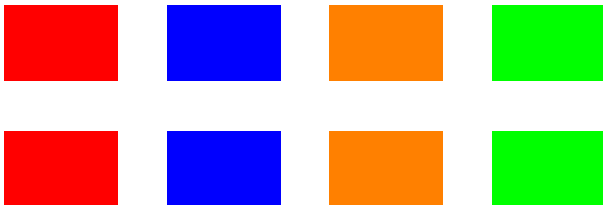
4 different architectures

on 0.18 μm CMOS ASIC

on FPGA (Virtex-4, Virtex-5, Virtex-II-Pro)

ASIC architectures

high-throughput: 8 ChaCha functions



BLAKE-64: 132 kGE 5.9 Gbps

BLAKE-32: 58 kGE 5.3 Gbps

ASIC architectures

small fingerprint: 1 ChaCha function



BLAKE-64: 20 kGE 181 Mbps

BLAKE-32: 10 kGE 253 Mbps

ASIC architectures

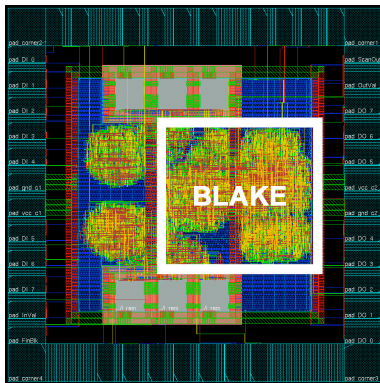
trade-off space/speed: 4 ChaCha functions



BLAKE-64: 82 kGE 4.8 Gbps

BLAKE-32: 41 kGE 4.1 Gbps

BLAKE-32 lightweight hashing core



13.5 kGE

128 Mbps

BLAKE in software (eBASH)

in the top 5

faster than SHA-2

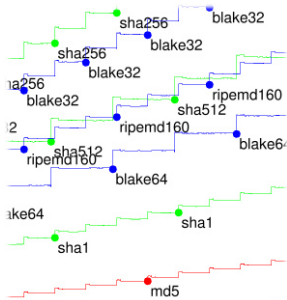
on NIST reference platform

BLAKE-64 \approx 10 cycles/byte

BLAKE-32 \approx 16 cycles/byte

eBASH figures

amd64, 2000MHz, AMD Athlon 64 X2 (40fb2), mace

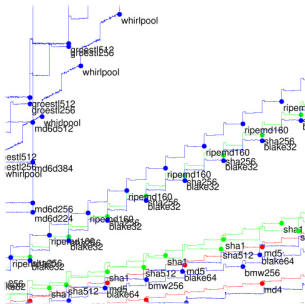


BLAKE-64 11.32 cycles/byte

BLAKE-32 17.86 cycles/byte

eBASH figures

ia64, 997MHz, HP Itanium II, nmi0020



BLAKE-64 8.54 cycles/byte

BLAKE-32 20.69 cycles/byte

why BLAKE for SHA-3 ?

one of the first candidates published, no attack
faster than SHA-2 and than many SHA-3 candidates
based on a known algorithm (ChaCha/Salsa20)

one of the simplest designs

no AES-dependence

BLAKE's webpage

<http://www.131002.net/blake/>

complete specification

security analysis

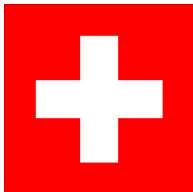
reference C code

eBASH submission

light C code

VHDL code

BLAKE



SWISS MADE