

Impossible Herd Collision with Unique HAIFA Counter Salting of Chaining Value

Introduction

In this preliminary [to be revised document section](#), we present FortressGB enhancements of two naïve implementations of the Biham/Dunkelman HAIFA Framework for new hash devices. In the ZK-Crypt III we developed a proven method for preventing Message modification. We developed a dual track feedback methodology whereby the feedback streams were orthogonal in the sense that each feedback stream affected different parts of the hash module, in a grossly unpredictable way. The provable result was that if the hacker found a way to reconcile the "damage" caused by one aberrated Message word, he would cause provably totally different immediately diffused irreconcilable changes in other interacting parts of the circuit. Thanks to the massive diffusion in the ZK-Crypts, where one false bit appears in the next clock in the equations of a minimum of 140 monomials further reconciliation is virtually impossible [zk-undst] [bard].

The 64 Bit HAIFA Counter outputs uniquely place each of the first (more than) 2^{61} chaining values, thereby preventing herded collisions. The outputs of the Mersenne Prime LFSRs are relatively prime, so that each of the LFSRs is outputs a pseudo random string.

Pseudo-randomly diffusing the 64 bit Hybrid Mersenne Prime LFSR/binary HAIFA counting mechanism into the two feedback streams, provably causes different irreconcilable feedbacks into 5 entry points into the Register Bank and the Data Processor. By randomly enumerating (marking) each chaining-value there cannot be conventional fix-points (identical chaining-values) in the sequence of more than 2^{61} unique counted Message words (more than 2^{66} message bits). A false 32 Bit Message obviously cannot rectify the new location 64 bit counts in the stages of a herd collision.

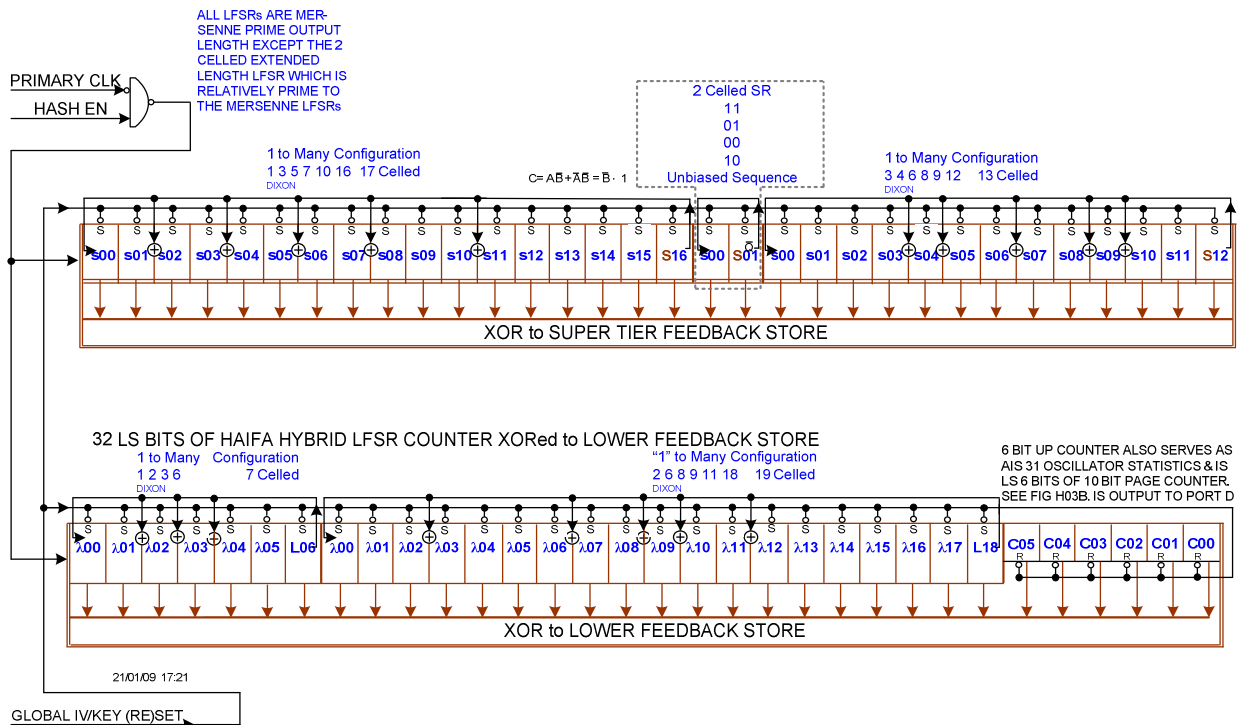


Fig. 1 The HAIFA counts are XOR combined to the Super Tier and Lower Feedback Streams

MAC MIX DISPLACEMENT

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 |
| INPUT | A | B | C | D | E | F | G | H | J | K | L | M | N | P | Q | R | S | T | U | V | W | X | Y | Z | a | b | c | d | e | f | g | h | |
| OUTPUT | D | C | B | A | H | G | F | E | M | L | K | J | R | Q | P | N | V | U | T | S | Z | Y | X | W | d | c | b | a | h | g | f | e | |

PERFORM THE MAC MIX - $f_{MMX}(NWR)$
TRANSFORM ON THE PRESENT RESULT

$M8=(88888888)_{16}$; $M4=(44444444)_{16}$
 $M2=(22222222)_{16}$; $M1=(11111111)_{16}$
 $MMM=0 \text{ MOD } 2^{32}$

INPUT NWR (32 BIT WORD)

$MMM=[(NWR \text{ AND } M8) / 8]$
 $MMM=[(NWR \text{ AND } M4) / 2] \text{ OR } MMM$
 $MMM=[(NWR \text{ AND } M2) \times 2] \text{ OR } MMM$
 $MMM=[(NWR \text{ AND } M1) \times 8] \text{ OR } MMM$
 $MMX=MMM$

RETURN MMX

THE MAC MIX DISPLACEMENT FUNCTION IS ACTIVATED IN DATA AUTHENTICATION & IN CIPHER MODE LOADING OF EXTENDED PORTION OF SECRET KEY & ALSO LOADING FULL LENGTH IV.

SUPERMIX DISPLACEMENT

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INPUT | A | B | C | D | E | F | G | H | J | K | L | M | N | P | Q | R | S | T | U | V | W | X | Y | Z | a | b | c | d | e | f | g | h |
| OUTPUT | d | c | b | a | h | g | f | e | D | C | B | A | H | G | F | E | M | L | K | J | R | Q | P | N | V | U | T | S | Z | Y | X | W |

GENERATE SUPERMIX - $f_{SMX}(ISX \oplus BSF)$,
AND BOTH MODES OF SUPER TIER FB

$M8=(88888888)_{16}$; $M4=(44444444)_{16}$
 $M2=(22222222)_{16}$; $M1=(11111111)_{16}$
 $MMM=0 \text{ MOD } 2^{32}$

INPUT ISX, BSF

$TMP=ISX \oplus BSF$

$MMM=[(TMP \text{ AND } M8) / 8]$
 $MMM=[(TMP \text{ AND } M4) / 2] \text{ OR } MMM$
 $MMM=[(TMP \text{ AND } M2) \times 2] \text{ OR } MMM$
 $MMM=[(TMP \text{ AND } M1) \times 8] \text{ OR } MMM$
 $SMX=8 \gg \gg MMM$; **SUPERMIX SALT**

IS CIPHER MODE ?

MAC MODE
INPUT MMX

$SUP=MMX \oplus SMX \oplus \text{DISPERSED HAIFA BITS}$

OUTPUT SUP TO SUPERTIER FB STORE

| | | | | | | | | | | | | | | | |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| J | K | L | M | N | P | Q | R | S | T | U | V | W | X | Y | Z |
| D | C | B | A | H | G | F | E | M | L | K | J | R | Q | P | N |

THE SUPERMIX DISPLACEMENT IS ACTIVATED BY THE PRIMARY CLK.

CIPHER MODE SUPER TIER FB CONSISTS OF SUPERMIX (SMX) \oplus DISPERSED "HAIFA" BITS.

MAC MODE SUPER TIER FB (SUP) = SUPERMIX \oplus MAC MIX \oplus "HAIFA" BITS

IN MAC MODE THE SUPERMIX SALTS THE REVERSED NIBBLE PRESENT RESULT (MMX)

$SUP = SMX \oplus \text{DISPERSED HAIFA}$

OUTPUT SUP TO SUPERTIER FB STORE

34SUMX

Fig. 2 The MAC MIX & MUX & SuperMIX SALT Displacement Permutations Feeding the Super Tier

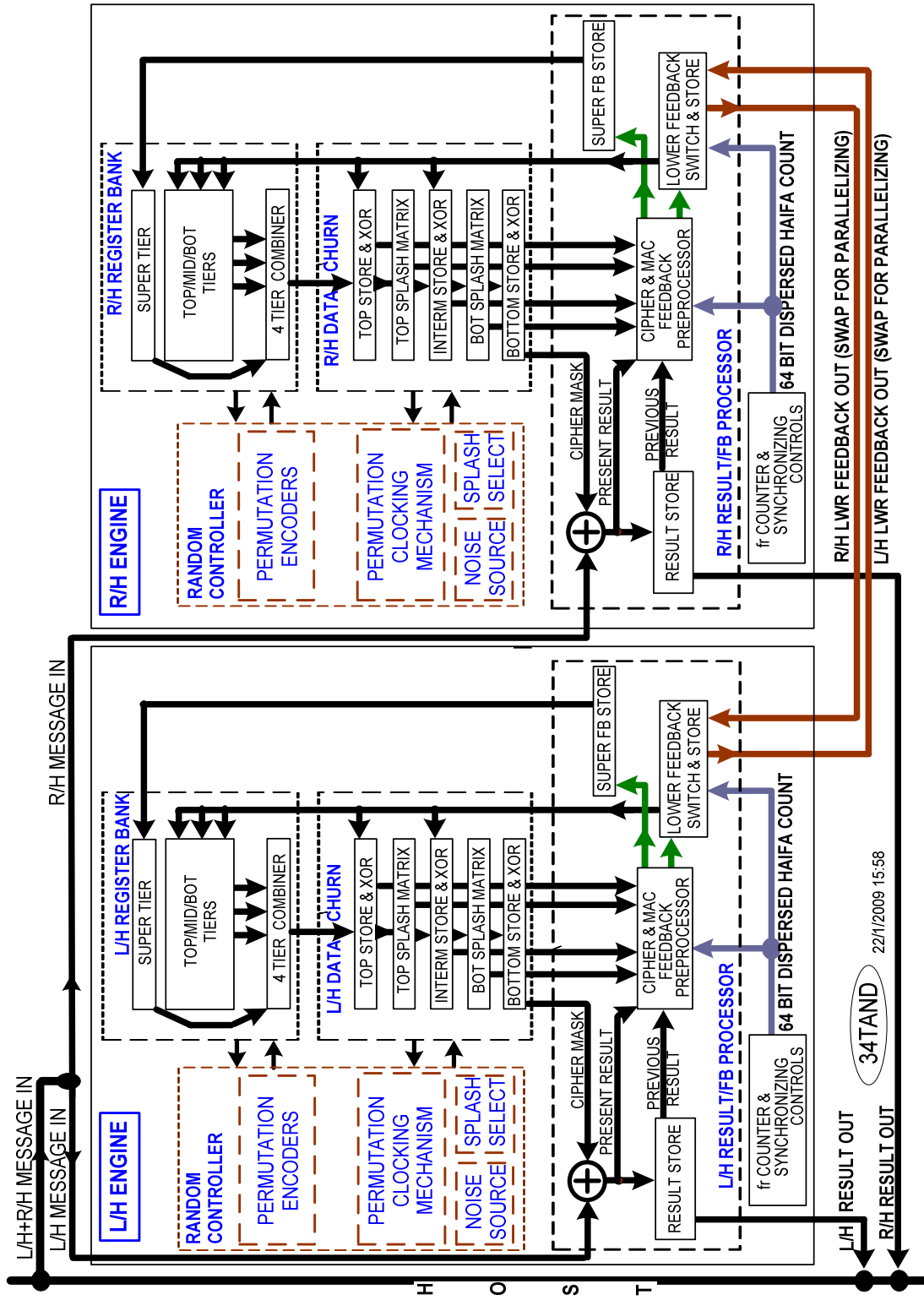


Fig. 3 Parallelized ZK-Crypt Autonomous Engines with Swapped Lower Feedback

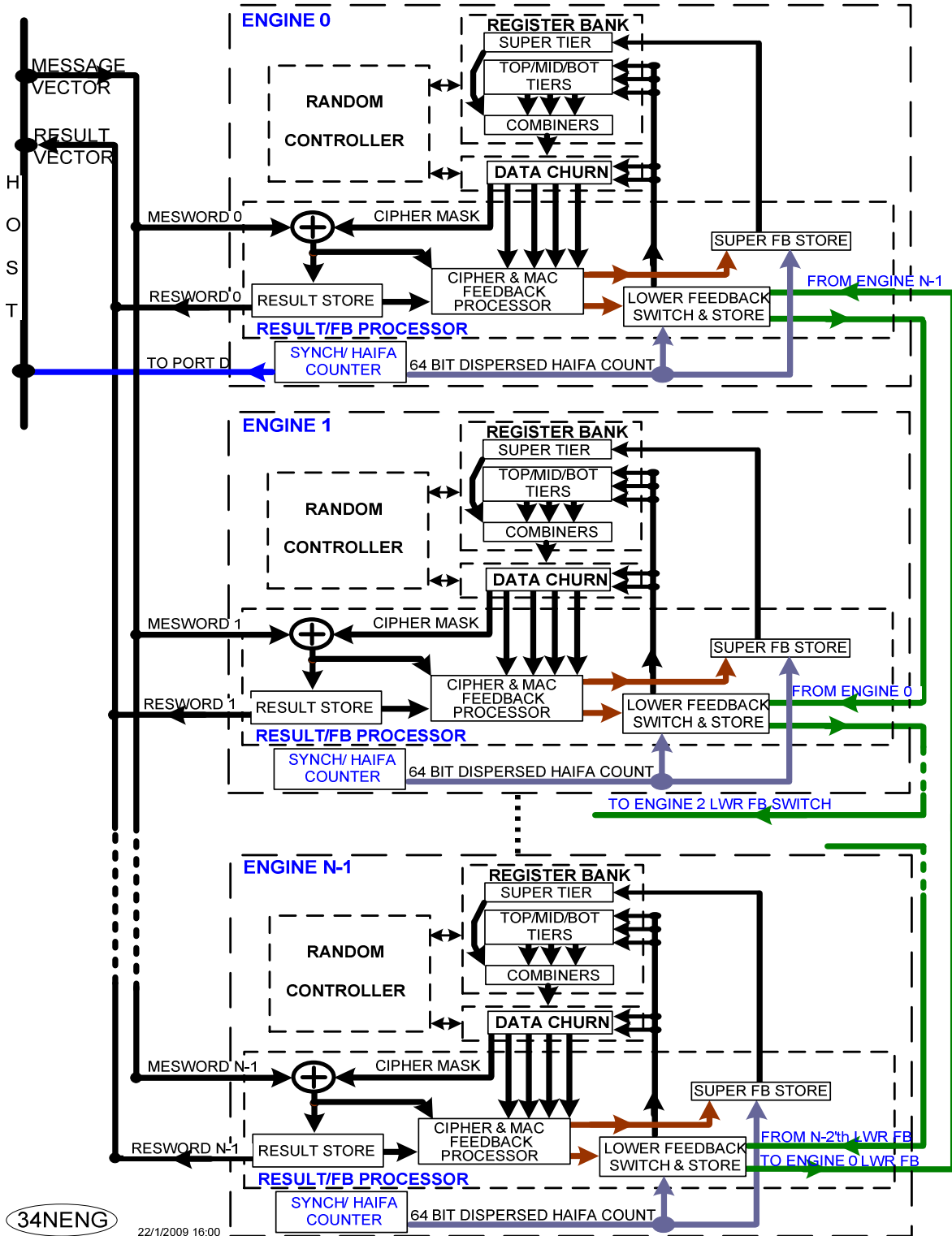


Fig. 4 N ZK-Crypt Cipher/Hash Autonomous Engines- i'th Lower FB into i+1'th mod N Engine

References:

- [bard] G. Bard, Algorithms for the Solution of Linear and Polynomial Systems of Equations over Finite Fields, with Applications to Cryptanalysis, PhD Thesis, U of Maryland, Dept. of Mathematics, 2007.
- [biham1] E. Biham, O. Dunkelman, A Framework for Iterative Hash Functions, NIST Hash Forum 2006
- [biham2] E. Biham, O. Dunkelman, Differential Cryptanalysis in Stream Ciphers, Technion CS Dept Tech Report CS-2007.
- [bernst] D.J. Bernstein, "Does the ZK-Crypt I flunk the repetition test", eSTREAM website, March, 2006.
- [cusick] T.Cusick, C.Ding & A.Renvall, Stream Ciphers & Number Theory, N. Holland Mathematical Library, Elsevier, 2004.
- [daemen] J.Daemen, Invited Lecture-Guidelines, SASC 2006 Stream Ciphers Revisited, Workshop Record, Leuven, February 2006.
- [kelsey] J. Kelsey, Invited Lecture, eCRYPT Hash Workshop 2007, Barcelona, May 2007.
- [sasc-eth] F. Gürkaynak et al, "Hardware Evaluation of eSTREAM Candidates", SASC 2006 Stream Ciphers Revisited, Workshop Record, Leuven, February 2006.
- [zk-algo] A. Hecht, O. Dunkelman, C. Gressel, The ZK-Crypt Algorithmic Specification, w FortressGB.com, April 2007.
- [zk-a-z glos] C. Gressel, O. Dunkelman, A. Hecht, The A-Z Guide to the ZK-Crypt, An annotated glossary, w fortressgb.com, vers 3, April 2007.
- [zk-ccc] C. Gressel, ZK-Crypt Circuit Concept Drawings, [w fortressgb.com](http://w.fortressgb.com), London & Omer, April 2007.
- [zk-code] A. Hecht, ZK-Crypt C Code Simulator, [w fortressgb.com](http://w.fortressgb.com), London & Omer, April 2007.
- [zk-fbpat] C. Gressel, O. Dunkelman, G. Bard, A. Hecht, R. Granot, A System & Method to Preclude Message Modification in Data Authentication..., PCT/IL/2007001101, Biham was also a co-inventor. He claimed his innovation was obvious, and is not listed.
- [zk-intrfac] "What the Host Sees in the ZK-Crypt", Interfacing the ZK-Crypt functions, [w fortressgb.com](http://w.fortressgb.com), vers January 2007.
- [zk-secure] G. Bard, C. Gressel O. Dunkelman, A. Hecht, The Expanded ZK-Crypt Security Analysis, w FortressGB.com, January 2008.
- [zk-undst] C.Gressel, O. Dunkelman, A. Hecht, "Understanding the ZK-Crypts- Ciphers for (Almost) all Reasons", w FortressGB.com, January 2008