Revisiting the IDEA Philosophy

Pascal Junod[1,2] Marco Macchetti[2]

[1]University of Applied Sciences Western Switzerland (HES-SO)

[2]Nagracard SA, Switzerland

FSE'09 Leuven (Belgium), February 24, 2009

Outline

The IDEA Block Cipher

Outline

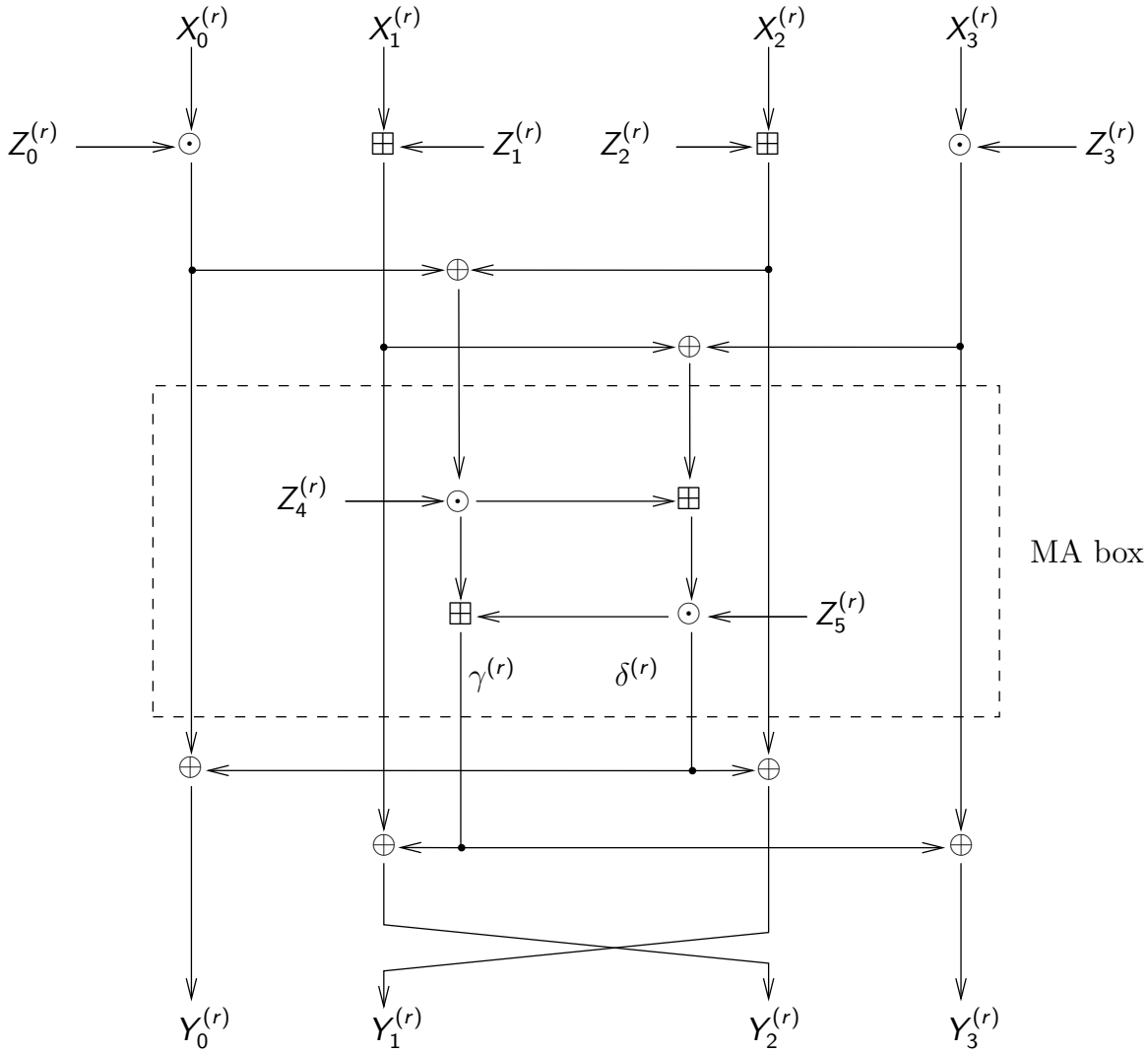
Outline

Outline

A Bit of History

- Designed by Lai and Massey in 1990 on behalf of ASCOM AG (the patent now belongs to Nagravision SA)
- Block cipher encrypting 64-bit blocks under a 128-bit key
- Very simple philosophy: mix three different and algebraically incompatible group laws on 16-bit values:
  - $\oplus$ = XOR
  - $\boxplus$ = addition modulo 2^{16}
  - $\odot$ = multiplication modulo $2^{16} + 1$
- Simple, fully linear bit-selecting key-schedule algorithm
- Quite popular during the 90's thanks to PGP

IDEA Round Function



IDEA is a Secure Cipher

- Designed to resist differential cryptanalysis
- Extensively cryptanalyzed (more than 15 published papers so far)
- Today, the best attack by Biham et al. [BDK07] breaks 6 rounds (out of 8.5) using the full codebook and within a complexity of $2^{126.8}$ operations in a classical scenario
- Virtually all the attacks largely exploit properties of the fully linear, bit-selecting key-schedule algorithm

IDEA will Fall into the Public Domain

- The "IDEA way" to build a cipher looks like to be valid in terms of security
- By the way, IDEA will fall into the public domain on May 16th, 2011
- Existing extensions (like MESH ciphers) are not very competitive in terms of speed
- Can we re-use this approach to design something new and fast (with a look at hash functions and authenticated encryption schemes) ?

Lipmaa's Implementation on the MMX Architecture

- At SAC'97, Lipmaa published the so far fastest implementation of IDEA on Intel CPUs
- Exploits the SIMD features of the MMX instruction set
- Obtained a 4-way implementation able to encrypt at ≈18 clocks/byte on Pentium and Pentium II CPUs
- Such implementations are useful to perform ECB, CBC decryption or CTR mode
- More parallelization lead to (hard-to-use-in-practice) bitslice implementations

A New Implementation with SSE*x* Instructions

- The Intel SSE*x* architecture brings up to 16 128-bit registers (XMM)
- Added full support for unsigned multiplication of 16-bit values performed 8-times in parallel
- On Core2 CPUs, the throughput and latencies of the SSEx instructions have been seriously improved

An 8-Way Branch-Free IDEA Multiplication

```
1   t  =  _mm_add_epi16    (a, b);
2   c  =  _mm_mullo_epi16  (a, b);
3   a  =  _mm_mulhi_epu16  (a, b);
4   b  =  _mm_subs_epu16   (c, a);
5   b  =  _mm_cmpeq_epi16  (b, XMM_0);
6   b  =  _mm_srli_epi16   (b, 15);
7   c  =  _mm_sub_epi16    (c, a);
8   a  =  _mm_cmpeq_epi16  (c, XMM_0);
9   c  =  _mm_add_epi16    (c, b);
10  t  =  _mm_and_si128    (t, a);
11  c  =  _mm_sub_epi16    (c, t)
```

An 8-Way Branch-Free IDEA Multiplication

```
1   t  =  (a + b) & 0xFFFF;
2   c  =  (a * b) & 0xFFFF;
3   a  =  (a * b) >> 16;
4   b  =  (c - a); if (b & 0x80000000) b = 0;
5   if (b == 0) b = 0xFFFF; else b = 0;
6   b  =  b >> 15;
7   c  =  (c - a) & 0xFFFF;
8   if (c == 0) a = 0xFFFF; else a = 0;
9   c  =  (c + b) & 0xFFFF;
10  t  =  t & a;
11  c  =  (c - t) & 0xFFFF;
```

Our implementation

- Implementation of 8-way IDEA on the `x86_64` architecture using SSE2 instructions
- Integrated our 8-way IDEA implementation in CTR mode into the eSTREAM benchmarking framework
- Our implementation is running at 5.4 clocks/byte on an Intel Core2 CPU (with a pre-computed key-schedule and for long messages).

Comparison with Other Ciphers

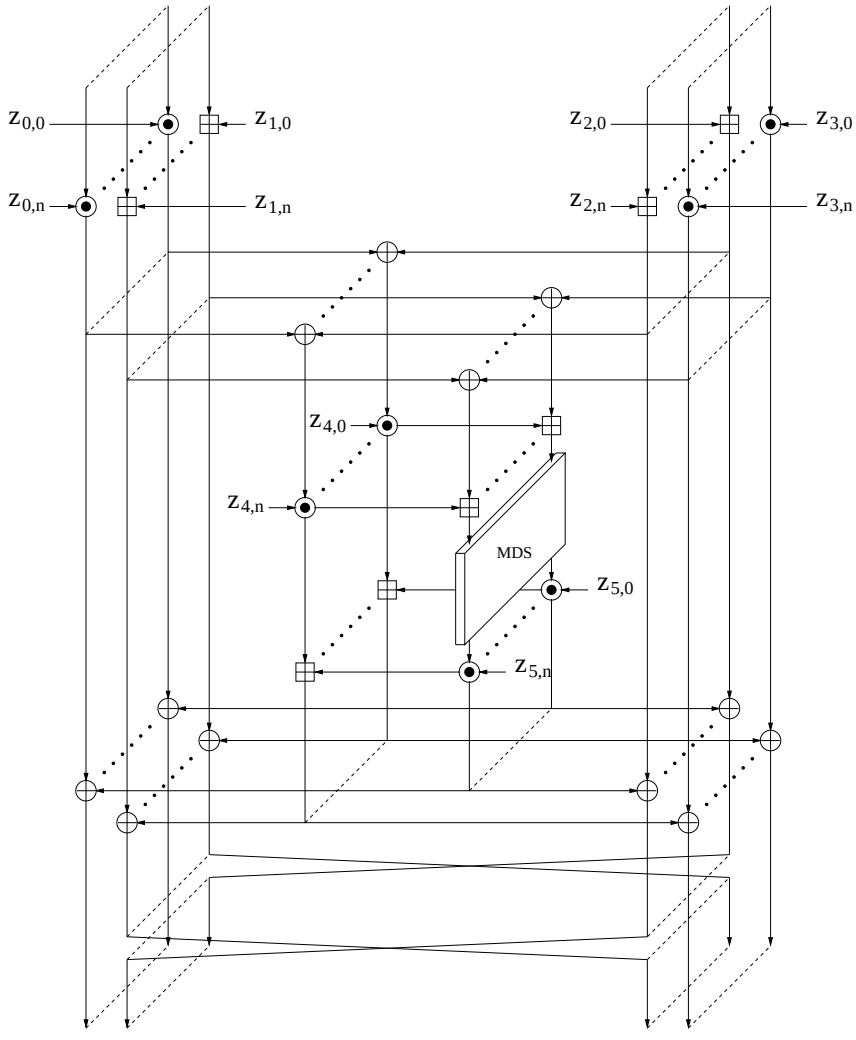
- Intel Core2 with long messages in clock/byte

Cipher	Speed	Source
Trivium	3.66	[eSTREAM-Bernstein]
Salsa 20/20	3.91	[eSTREAM-Bernstein]
IDEA-8	5.42	This paper
LEX v2	5.83	[eSTREAM-Bernstein]
RC4	7.47	[eSTREAM-Bernstein]
AES-128	9.20	(bitslice) [Matsui-Nakajima-2007]
AES-128	10.57	[Bernstein-Schwabe-2008]
AES-128	12.59	[eSTREAM-Bernstein]

Our Goals

- Build a block cipher with a $(64 \times n)$-bit block size
- Fully respect the IDEA design philosophy
- Design a new key-schedule algorithm
- Keep the highest possible parallelism (and speed)
- If possible, inherit all the good security properties of IDEA

WIDEA as a Single Picture



MDS Matrix

- $N \times N$ matrix over $\mathrm{GF}(2^{16})$ building an (N, N)-multipermutation
- Only step which is "somewhat" sequential (using pre-computed tables would be to expensive in terms of time)
- Still possible to perform the `xtime()` operation 8-times in parallel.

Key-Schedule Algorithm

- Non-linear feedback shift register
- Fast diffusion (full diffusion after 3 rounds of WIDEA)
- Asymmetry brought through iteration-dependent constants
- Design approach similar to the AES key-schedule

$$Z_i = ((((Z_{i-1} \oplus Z_{i-8}) \overset{16}{\boxplus} Z_{i-5}) \overset{16}{\lll} 5) \lll 24) \oplus C_{\frac{i}{8}-1}$$

Preliminary Security Considerations

- Two sequential operations are always algebraically incompatible
- Thanks to the MDS matrix , we get full diffusion after a single round
- Total of eight full diffusions (large number compared to other designs)
- Differential, linear and integral properties behave the same way than for IDEA
- We expect that the new, non-linear key-schedule further strengthens our design

WIDEA-8

- Fully specified in the paper, test vectors available
- 512-bit block size, 1024-bit key size
- WIDEA-8-based compression function in Davies-Meyer mode implemented as a Merkle-Damgard scheme on an Intel Core2 CPU using SSE3 instruction set

Speed Results

Cipher	Speed	Source
EDON-R 512	2.29	[NIST-EDONR]
WIDEA-8	5.98	This paper
CubeHash8/32	6.03	[NIST-CUBEHASH]
Skein-512	6.10	[NIST-SKEIN]
Shabal-512	8.03	[NIST-SHABAL]
LUX	9.50	[NIST-LUX]
Keccak	10.00	[NIST-KECCAK]
BLAKE-64	10.00	[NIST-BLAKE]
Cheetah	13.60	[NIST-CHEETAH]
Aurora	26.90	[NIST-AURORA]
Grostl	30.45	[NIST-GROSTL]
ECHO-SP	35.70	[NIST-ECHO]
SHAvite-3	38.20	[NIST-SHAVITE]
Lesamnta	51.20	[NIST-LESAMNTA]
MD6	52.64	[EBASH]
ECHO	53.50	[NIST-ECHO]
Vortex	56.05	[NIST-VORTEX]
FUGUE	75.50	[NIST-ECHO]

Future Work

- Fully specify a hash function (might be useful if NIST is unable to select a secure and fast SHA3 winner or AES-based constructions are broken ;-)
- Break it or prove that breaking it implies an attack on IDEA