Minalpher

DIAC 2014

Yu Sasaki[1], Yosuke Todo[1], Kazumaro Aoki[1],
Yusuke Naito[2], Takeshi Sugawara[2], Yumiko Murakami[2],
Mitsuru Matsui[2], Shoichi Hirose[3]

1: NTT 2: Mitsubishi Electric 3: Fukui University

How to pronounce

Minalpher [mɪnˈælfə]

Alpha → Alpher → Min-alpher → Minalpher

Minalpher is already a winner in the categories of…

Longest Name: 9 chars

with AVALANCHE, Enchilada and Raviyoyla

Longest Document: 70 pages

we designed everything from scratch

Minalpher: Design Concepts
Easy to Use in Practice

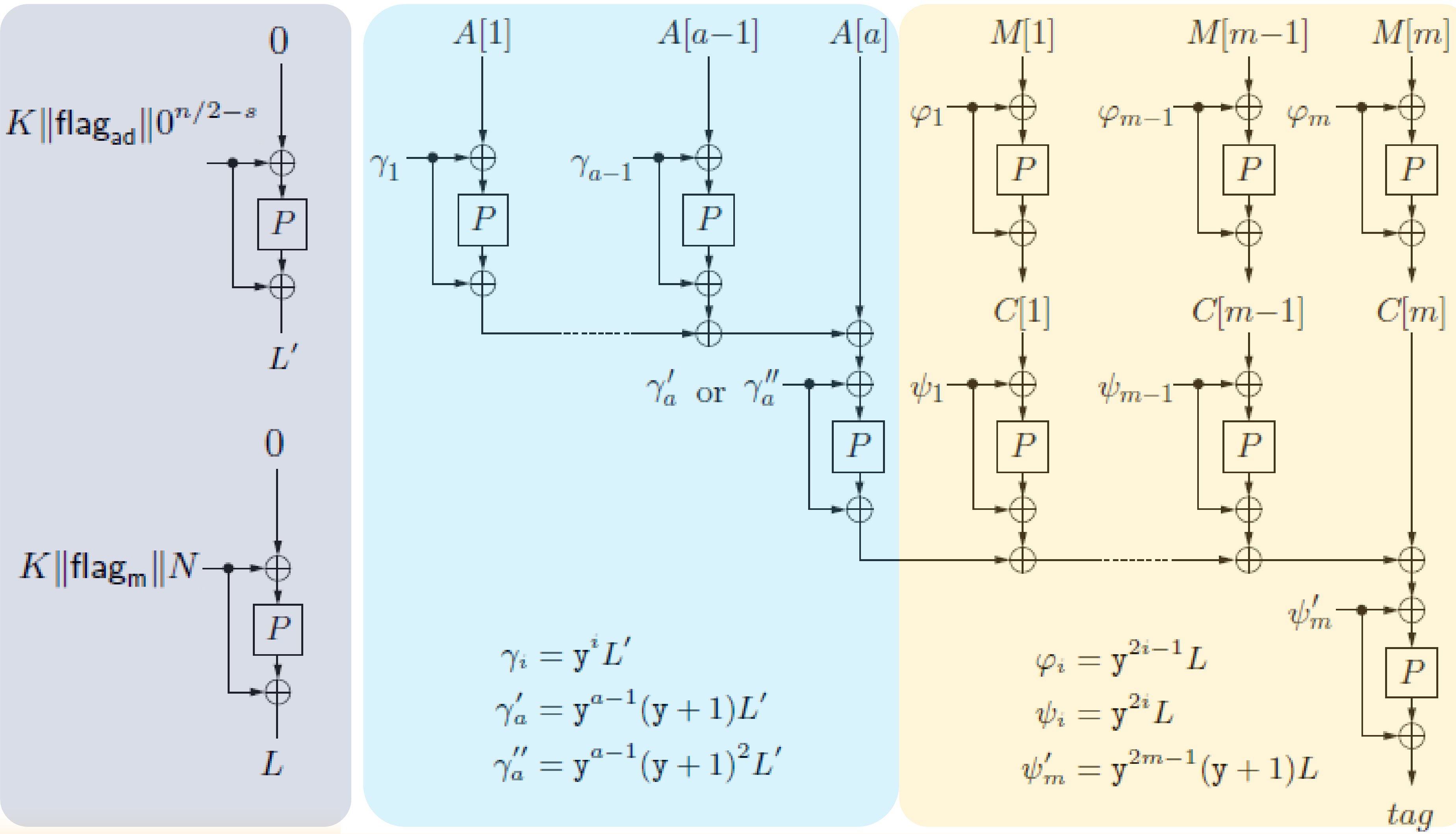
- **128-bit security (with <u>256-bit</u> permutation)**
  - 128-bit confidentiality
  - 128-bit authenticity
- **Additional security in misuse scenarios**
  - nonce repetition (nonce misuse)
  - release of unverified plaintext (decryption misuse)

Minalpher: Design Concepts
Easy to Use in Practice

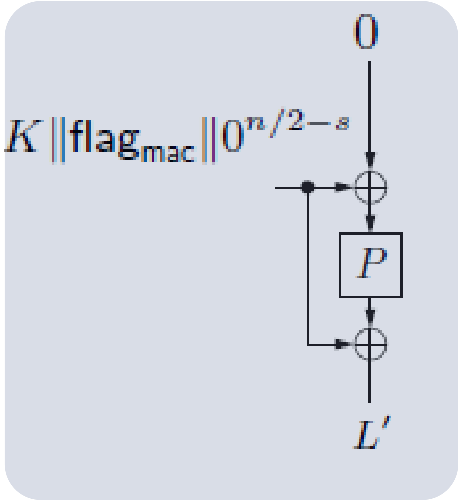
- **One algorithm for various platforms**
  - Fully parallelizable mode (fast on high-end platforms)
  - Simple, repetitive structure (small on embedded systems)
- **Additional functionalities**
  - MAC-only mode (faster than ciphertext-discarding AEAD)
  - Associated data reuse (faster 2nd time and afterwards)
  - Incremental computation (faster in nonce reuse scenario)
- **No Patent Submitted**

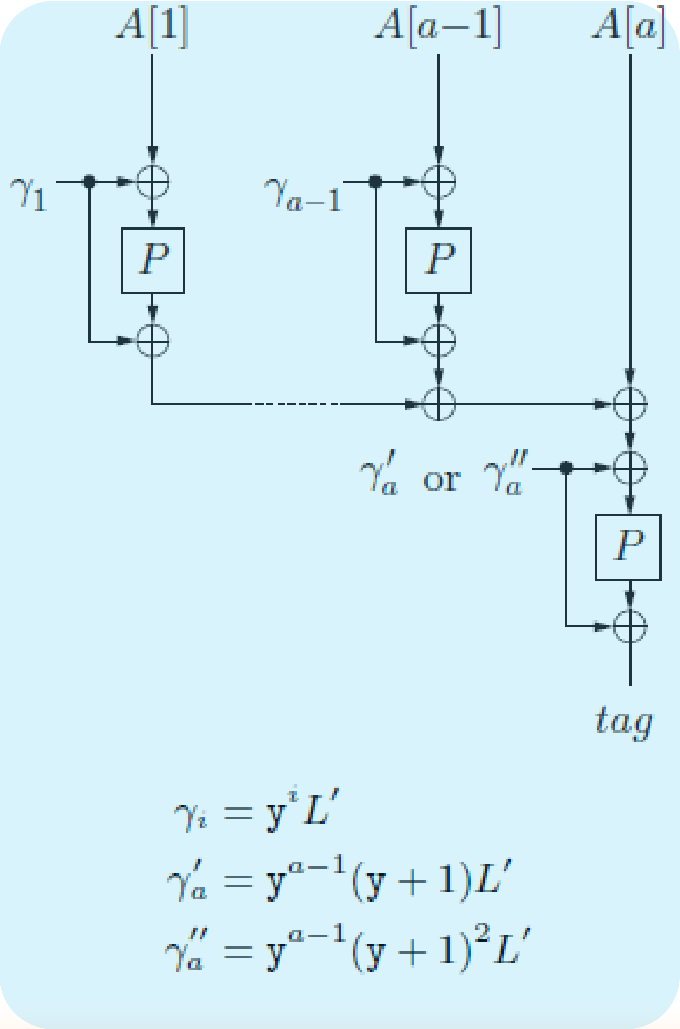
DESIGN

Minalpher (AEAD mode): Overview



Minalpher (MAC mode): Overview





y denotes a root of

$$Y^{32}+Y^3+Y^2+\mathbf{x}=0,$$

*where **x** is a root of*

$$X^8+X^7+X^5+X+1=0$$

Design Parameters

Key Size	128 bits
Nonce Size	104 bits
Tag Size	128 bits
Block Size	256 bits

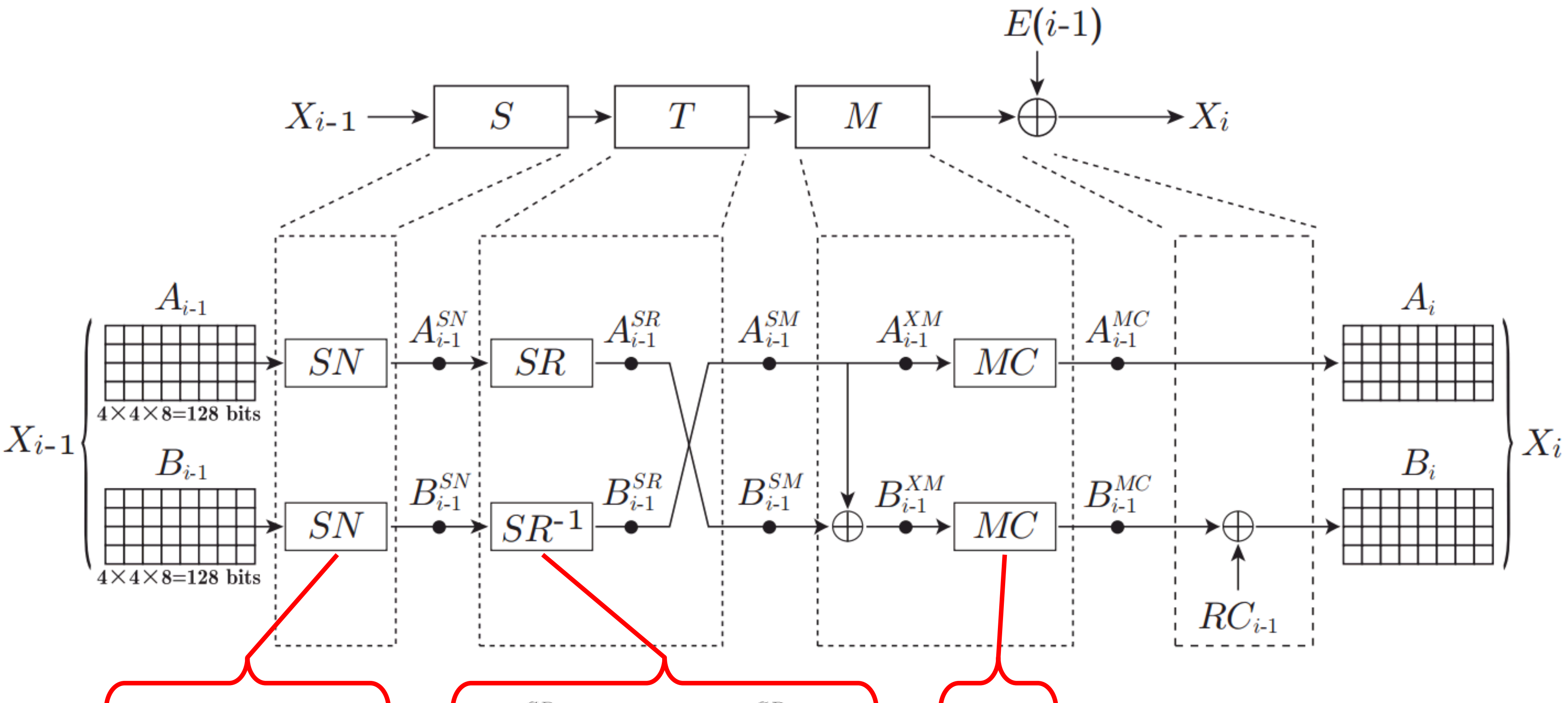
Max Size of AD+MSG in the AEAD mode	$2^{104} - 1$ bits
Max Size of MSG in the MAC mode	$2^{104} - 1$ bits

(secret message number is not supported in Minalpher)

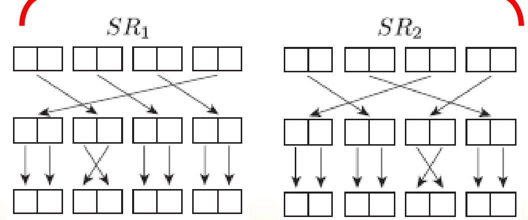
Minalpher-P: 256-bit Permutation

- **Nibble-wise (4-bit) Architecture**
  - 1 block = 256 bits = 64 nibbles
- **17.5-round Involutive SPN Structure**
  - *SN* (SubNibble): An Involutive 4-bit S-box
  - *SR* (ShuffleRows): Byte shuffle + Nibble swap
  - *MC* (MixColumns): A Binary 4x4 Matrix
- **$P = P^{-1}$ except round constants**

Minalpher-P: One Round



0xB	0x3	0x4	0x1
0x2	0x8	0xC	0xF
0x5	0xD	0xE	0x0
0x6	0x9	0xA	0x7



$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

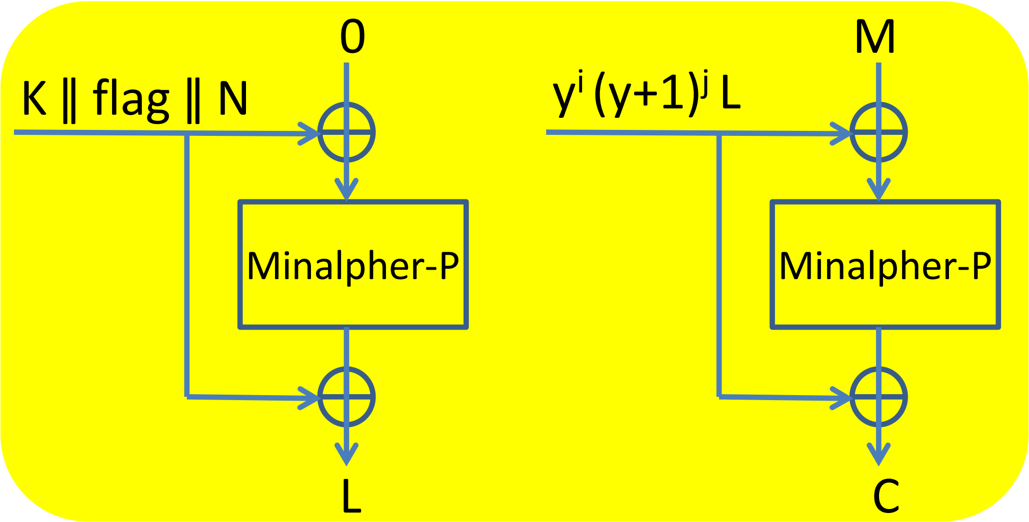
SECURITY

Security of Tweakable Even-Mansor

- **Tweakable Even-Mansor (TEM) is a 128-bit Strong Tweakable Pseudorandom Permutation (STPRP) in the ideal permutation model.**

$$\mathrm{Adv}^{\mathrm{stprp}}_{\mathsf{TEM}}(\mathcal{D}) = \Pr\left[K \xleftarrow{R} \mathcal{K} : \mathcal{D}^{\mathsf{TEM\_Enc}_K, \mathsf{TEM\_Dec}_K, \mathsf{P}, \mathsf{P}^{-1}} \Rightarrow 1\right] - \Pr\left[\mathcal{D}^{\mathsf{TRP\_F}, \mathsf{TRP\_B}, \mathsf{P}, \mathsf{P}^{-1}} \Rightarrow 1\right]$$

$$\leq \frac{\sigma^2}{2^{n-1}} + \frac{\sigma}{2^{n/2}}$$



Tweakable Even-Mansor Encryption with Minalpher -P

Security of Mode of Operation

- **Minalpher achieves 128-bit security for both privacy and authenticity**

  - Privacy: IND-CPA

$$\mathrm{Adv}^{\mathrm{priv}}(A) = \Pr[K \leftarrow K : A^{E_K} \Rightarrow 1] - \Pr[A^{\$} \Rightarrow 1]$$
$$\leq \mathrm{Adv}^{\mathrm{tprp}}_{\mathrm{TEM}}(D) + \frac{\sigma^2}{2^{n+1}} + \frac{q^2}{2^n}$$

  - Authenticity: INT-CCA

$$\mathrm{Adv}^{\mathrm{auth}}(A) = \Pr[K \leftarrow K : A^{E_K, D_K} \text{forges}]$$
$$\leq \mathrm{Adv}^{\mathrm{stprp}}_{\mathrm{TEM}}(D) + \frac{q}{2^l} + \frac{\sigma^2}{2^{n+1}} + \frac{q^2}{2^n}$$

Security in Misuse Scenarios

- **Minalpher achieves full authenticity and some privacy even in the following misuse scenarios:**
  – Release of unverified plaintext (RUP) , Nonce repetition (NR)

Authenticity

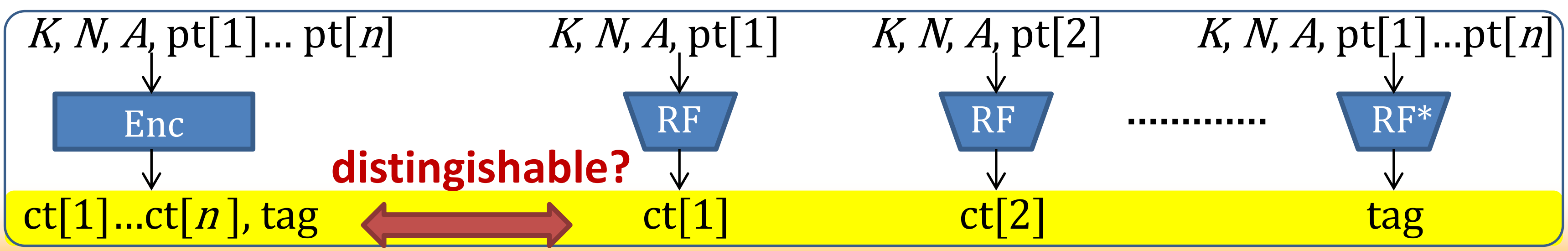
	¬RUP	RUP
¬NR	128-bit INT-CCA	128-bit INT-CCA
NR	128-bit INT-CCA	128-bit INT-CCA

Privacy

	¬RUP
¬NR	128-bit IND-CPA
NR	128-bit BW-PRF

BW-PRF: Block-wise pseudo-random function

Blocks ct[i] are indistinguishable from RF(K, N, A, pt[i])



Security against Various Cryptanalysis

- **128-bit key + 256-bit block**
  - The structure prevents a class of cryptanalysis requiring at least 2^{128} cost, e.g. MitM attacks, rectangle attacks.
- **Enough Security Margin**
  - Differential/linear characteristic probability at most 2^{-128} in 7 rounds out of the full 17.5 Minalpher-P rounds.
  - No 12-round attacks of Minalpher(-P) detected so far, e.g. boomerang attacks, amplified boomerang attacks, integral attacks, impossible differential attacks, truncated differential attacks, rebound attacks, etc.

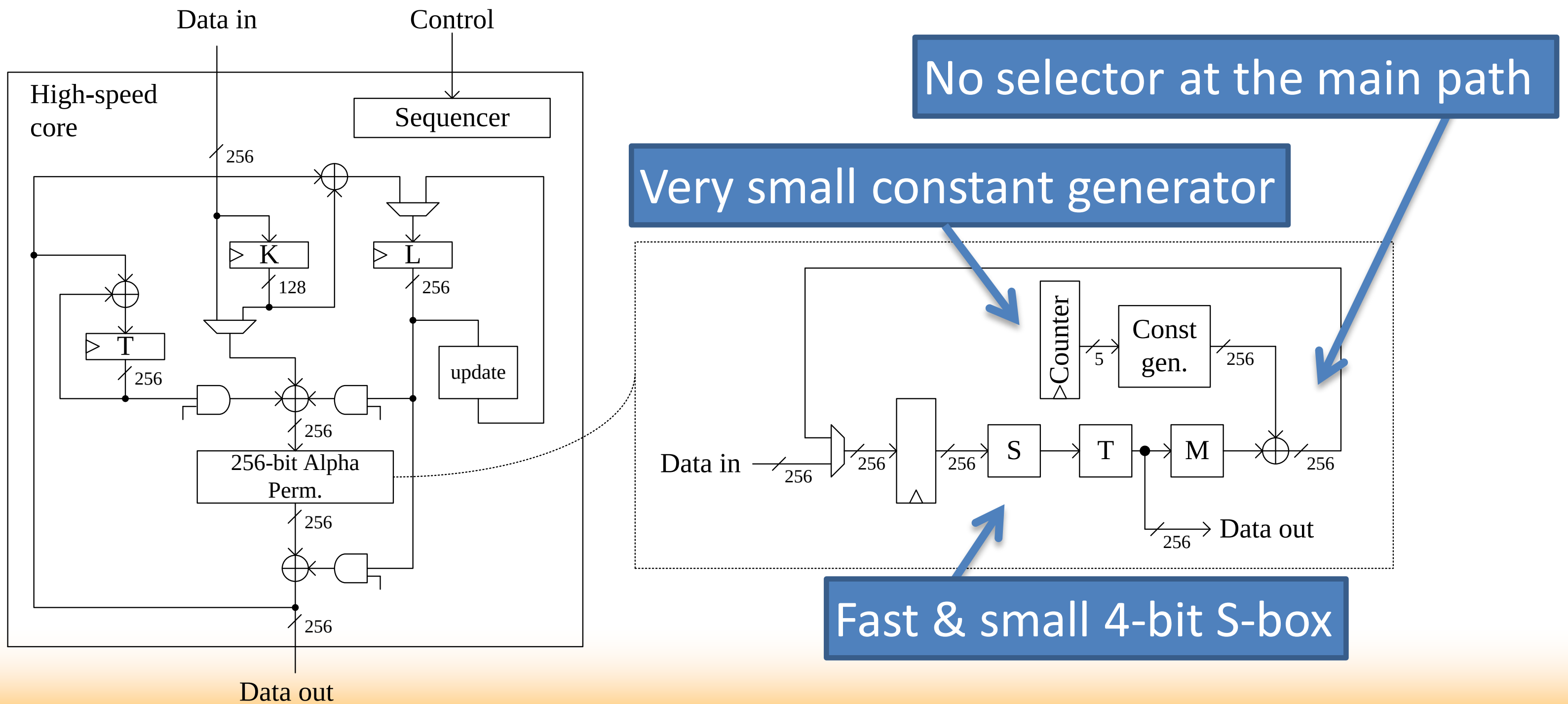
PERFORMANCE

Hardware implementation

- **Small S-box (4x4) and regular structure enable efficient and scalable Minalpher-P circuits**
  - No need for a key-scheduling circuit
  - Small S-box based design common in lightweight crypto
  - Involutive property minimizes the number of selectors
- **Three different hardware architectures are shown in the document**
  - High-speed core, mid-range core and low-area coprocessor
  - Evaluated with an open-source library: NanGate 45-nm CMOS
- **Further high throughput is possible if parallelized**

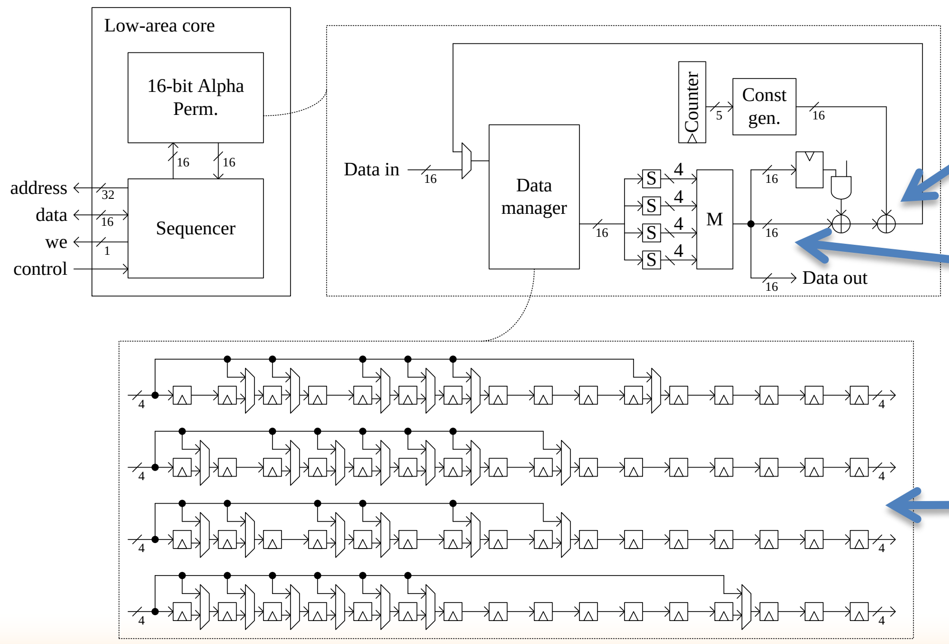
High-Speed Core

	Area [kGE]	Throughput [Mbps]
Minalpher-P Enc. and Dec., 1-round/cycle	4.96	7,771.71
AES Enc. Only, 1-round/cycle	10.49	1,587.50
Minalpher	14.32	6,103.96



Low-Area Coprocessor

	Area [kGE]	Throughput [Mbps]
Minalpher-P, Enc. and Dec., 16-bit datapath	2.70	375.06
AES Enc. only, 8-bit datapath	3.71	50.52
Minalpher	2.81	369.34



Almost no selectors at the main path

Fast & Small 4-bit S-box (again)

Complexity of ShuffleRows can be absorbed in the shift-register layer

Minalpher on Intel 64 Architecture

- Minalpher is designed to be well-suited on Intel 64 platform.
- The vpshufb instruction works very efficiently on SN (SubNibbles) and SR (ShuffleRows).
- Parallel block implementation can achieve faster speed.

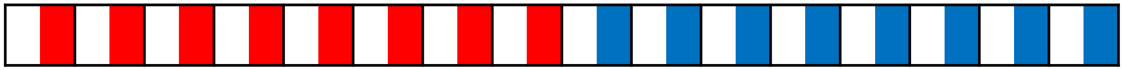
Processor	Implementation Method	Data Length / Cycles per byte				
		31B	63B	1KB	8KB	64KB
Core i7-3770 (Ivy Bridge)	1-block	23.1	19.1	14.6	14.4	14.4
Core i7-3770 (Ivy Bridge)	2-block parallel	23.4	16.5	10.0	9.6	9.6
Core i7-4770 (Haswell)	4-block parallel	---	---	---	---	6.3*

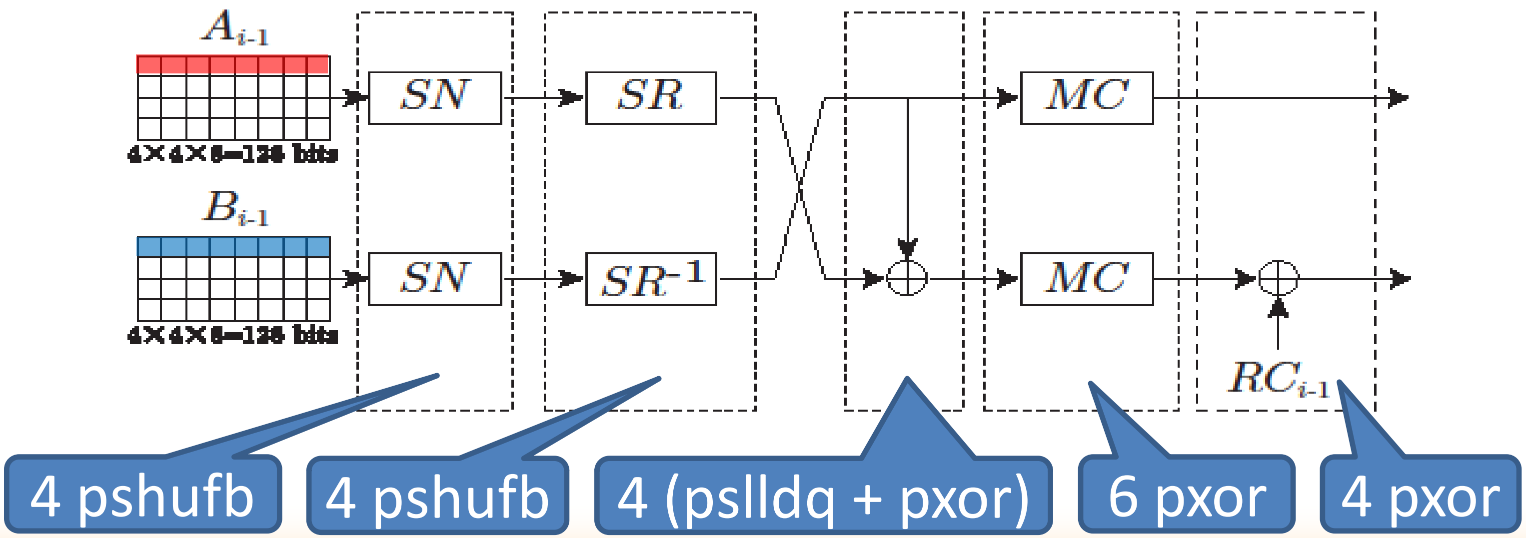
*Estimation based on the implementation of Minalpher-P

Round Function on Ivy/Sandy Bridge (1-block implementation)

Each nibble is stored in an octet of an XMM register.
Each register contains both rows of A and B.

XMM Register





A Code Example of One Round

State-In

SN:
```
vpshufb xmm0, xmm15, xmm0
vpshufb xmm1, xmm15, xmm1
vpshufb xmm2, xmm15, xmm2
vpshufb xmm3, xmm15, xmm3
```

SR:
```
vpshufb xmm0, xmm0, xmm14
vpshufb xmm1, xmm1, xmm13
vpshufb xmm2, xmm2, xmm12
vpshufb xmm3, xmm3, xmm11
```

XOR:
```
vpslldq xmm4, xmm0, 8
vpslldq xmm5, xmm1, 8
vpslldq xmm6, xmm2, 8
vpslldq xmm7, xmm3, 8
pxor    xmm4, xmm0
pxor    xmm5, xmm1
pxor    xmm6, xmm2
pxor    xmm7, xmm3
```

MC:
```
vpxor   xmm8, xmm4, xmm5
vpxor   xmm0, xmm8, xmm7
vpxor   xmm1, xmm8, xmm6
vpxor   xmm8, xmm6, xmm7
vpxor   xmm2, xmm8, xmm5
vpxor   xmm3, xmm8, xmm4
```

RC:
```
pxor    xmm0, [RC(r)0]
pxor    xmm1, [RC(r)1]
pxor    xmm2, [RC(r)2]
pxor    xmm3, [RC(r)3]
```

State-Out

S-box table is stored in xmm15

SR table is stored in xmm11-14

Minalpher on Low-end Microcontrollers

- **Minalpher is designed to be implemented in a small footprint on low-end microcontrollers**
  – The architecture Minalpher(-P) is simple and repetitive
  – A single involutional 4-bit S-box
  – Multiplying “y” in the tweak update is simple (3 byte-xors and x*2 in GF(2^8))
- **High speed implementation is also possible**
  – Two adjacent 4-bit S-boxes can be regarded as an 8-bit lookup table, which significantly improves performance.

Implementation Results

- **Target processor: RL78 (CISC microcontroller)**
- **Two Implementations**
  - **Small: minimizing ROM size**
  - **Fast: maximizing speed**

Design Goal	ROM (Bytes)	RAM (Bytes)	Speed (cycles)		
			Init	AD	Enc / Dec
Small	510	214	90,235	45,302	90,992/91,081
Fast	1,275	470	16,805	8,166	16,447/16,669

Init: Initialization (computing L and L')
AD: Processing of an associate data block
Enc/Dec: Processing of an encryption/decryption block

Conclusions

Minalpher: Easy to use with simple and unique design

- 128-bit security for privacy and authenticity
- Supporting MAC mode
- Security in misuse scenarios
- Fixed associate data reuse
- Incremental AE/MAC
- Fully parallelizable
- Lightweight tweak generation
- Involutive permutation
- Small S-box

Thank you!