How Fast Can a Two-Pass Mode Go? A Parallel Deterministic Authenticated Encryption Mode for AES-NI

Kazumaro Aoki   Tetsu Iwata   Kan Yasuda

NTT   Nagoya Univ   NTT

July 5, 2012 @ DIAC

Contents

- Background and design rationale
- New mode
- Security
- Implementation issues
- Conclusion

Background: Mode construction

- Recent blockcipher-based modes try to reduce $\mathrm{GF}(2^n)$ arithmetics (other than the blockcipher calls). The effect is very limited.
- Intel announced AES-NI, and a current standard PC can use AES-NI.
- Before AES-NI, CTR is slower than 7 cpb, and CTR is faster than 1 cpb after AES-NI.
- Even reducing the number of XORs becomes effective on AES-NI environment.

Background: NAE and DAE

- Authenticated encryption (AE) schemes has much attention. An AE should be used instead of using encryption only schemes. Recent attacks regarding BEAST and XML encryption might be avoided.
- AE modes can be classified into two categories: Nonce-based AE (NAE) and Deterministic AE (DAE). From the viewpoint of misuse-resistant, DAE is more preferable than NAE.

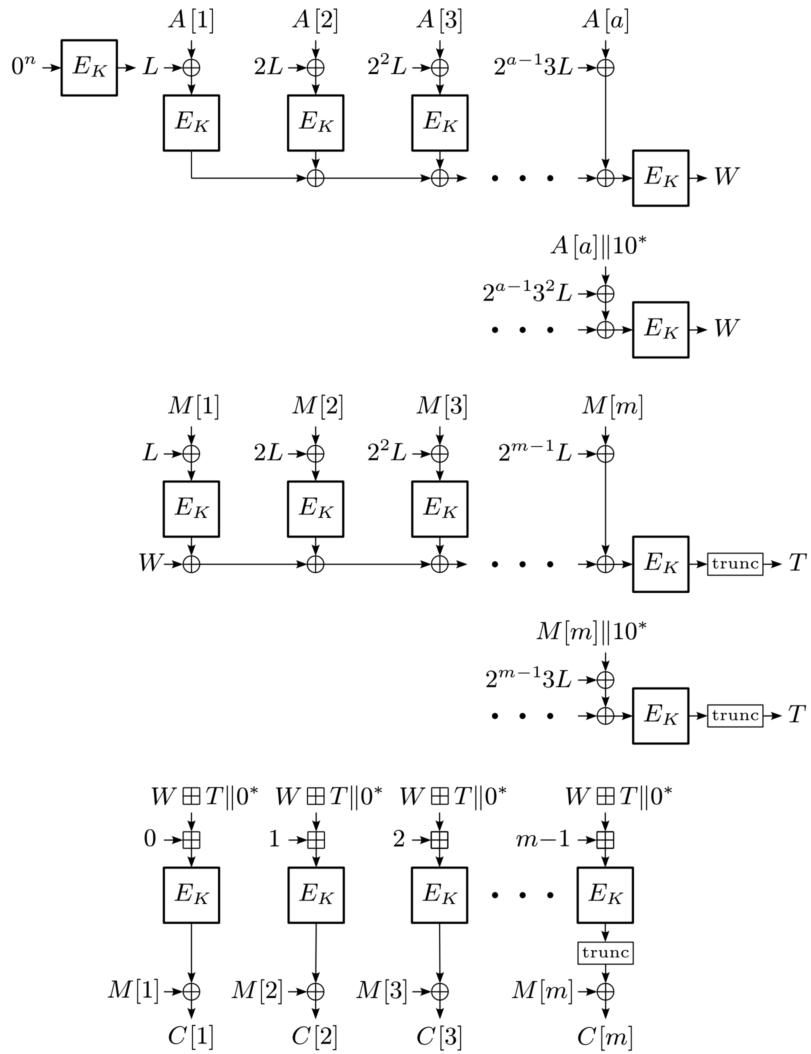
Background: Polynomial hash

- **A polynomial hash can be used for integrity, e.g., GCM and BTM.**
- **Intel also provides PCLMUL instruction, but it is not faster than expected.**
- **According to the tutorial at Indocrypt 2011, CTR performs 0.83 cpb, while GCM performs 2.59 cpb on Intel Sandy Bridge.**

Question: How fast can a DAE go, if the DAE is fully parallelizable and entirely blockcipher-based?

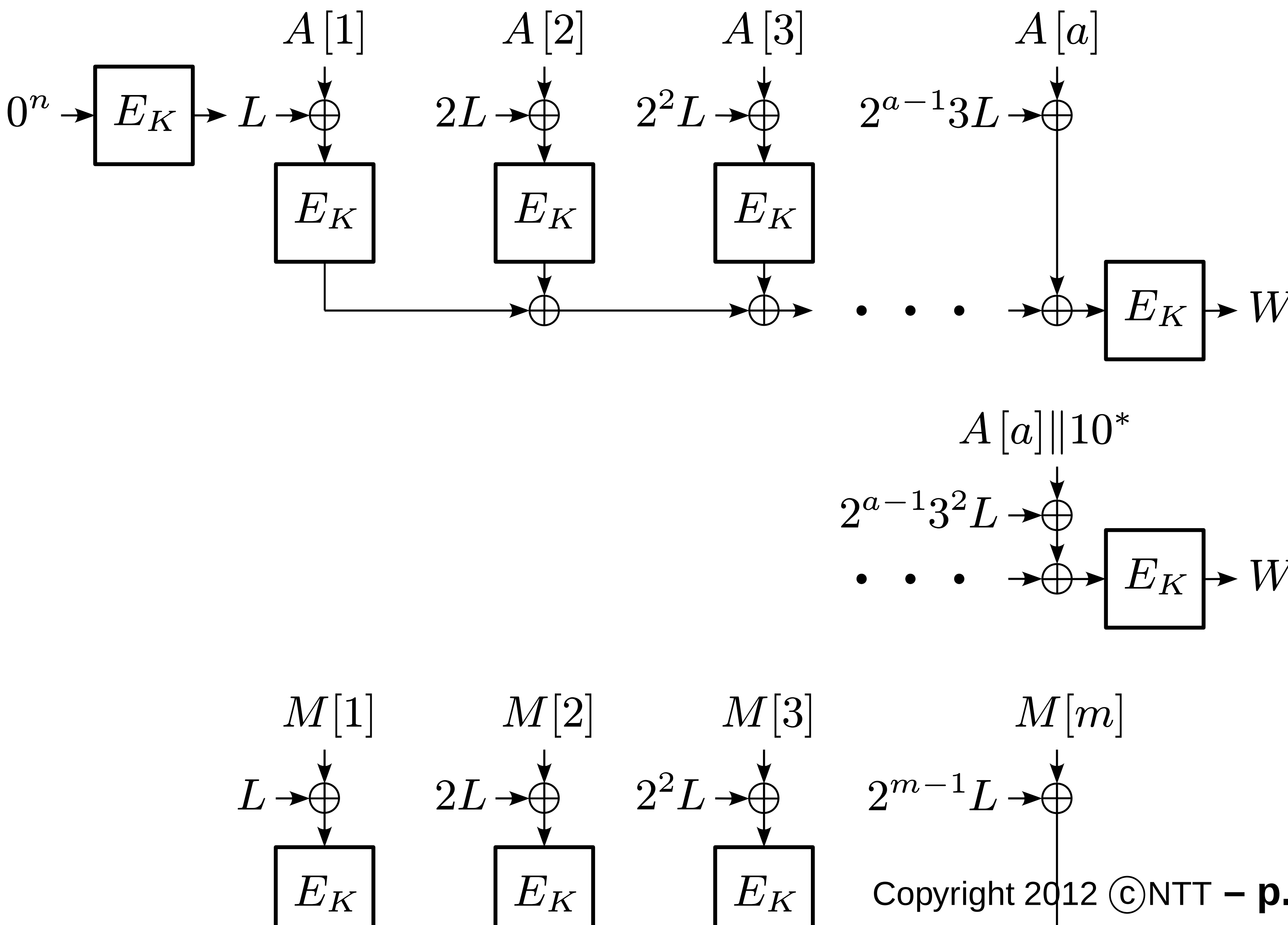
Our answer: PMAC + CTR

A new mode

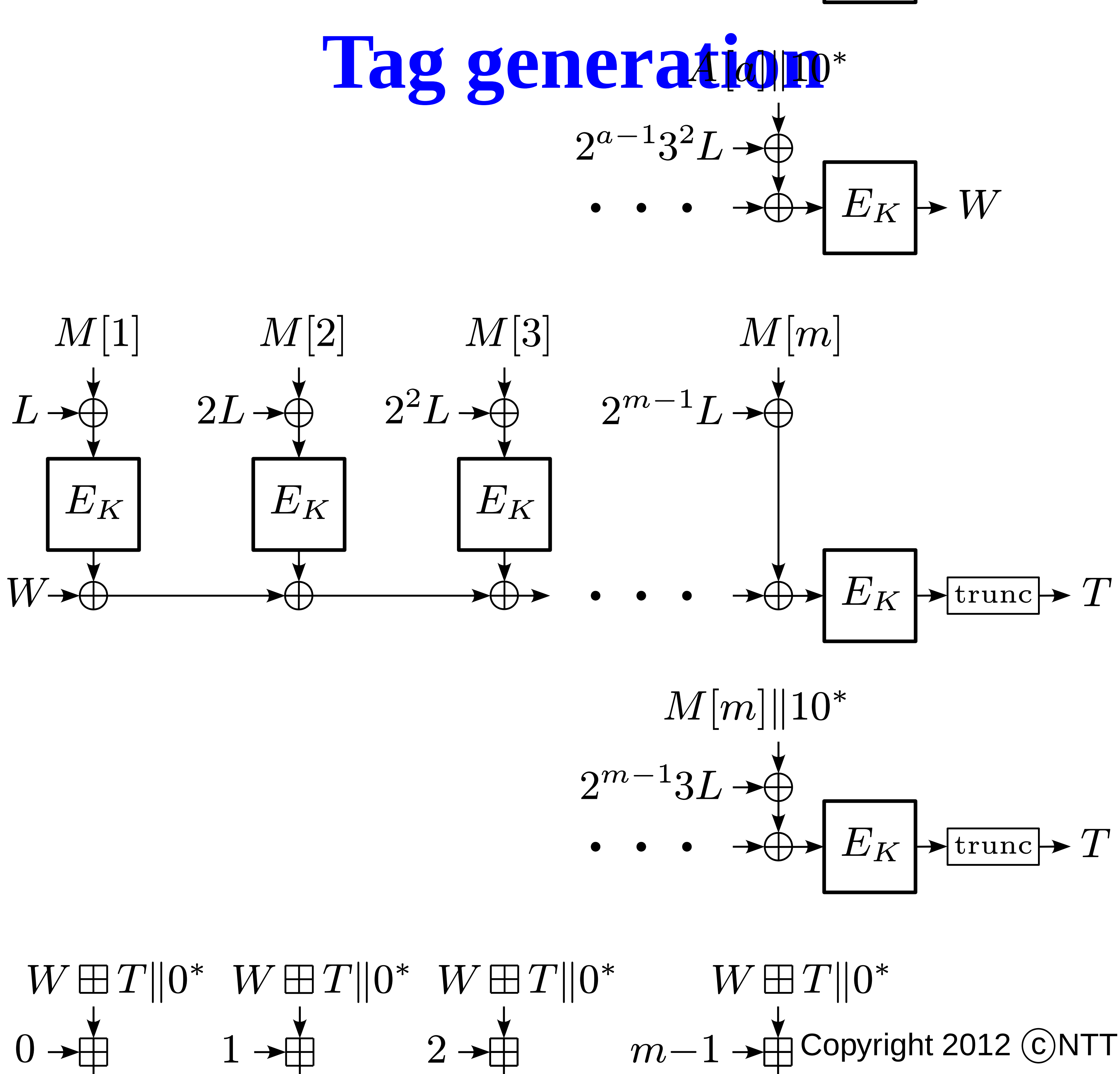


$A[1]$
$A[2]$
$A[3]$
$A[a]$
0^n
E_K
L
$2L$
2^2L
$2^{a-1}3L$
W
$A[a]\|10^*$
$2^{a-1}3^2L$
$M[1]$
$M[2]$
$M[3]$
$M[m]$
$2^{m-1}L$
trunc
T
$M[m]\|10^*$
$2^{m-1}3L$
$W \boxplus T\|0^*$
0
1
2
$m-1$
$C[1]$
$C[2]$
$C[3]$
$C[m]$

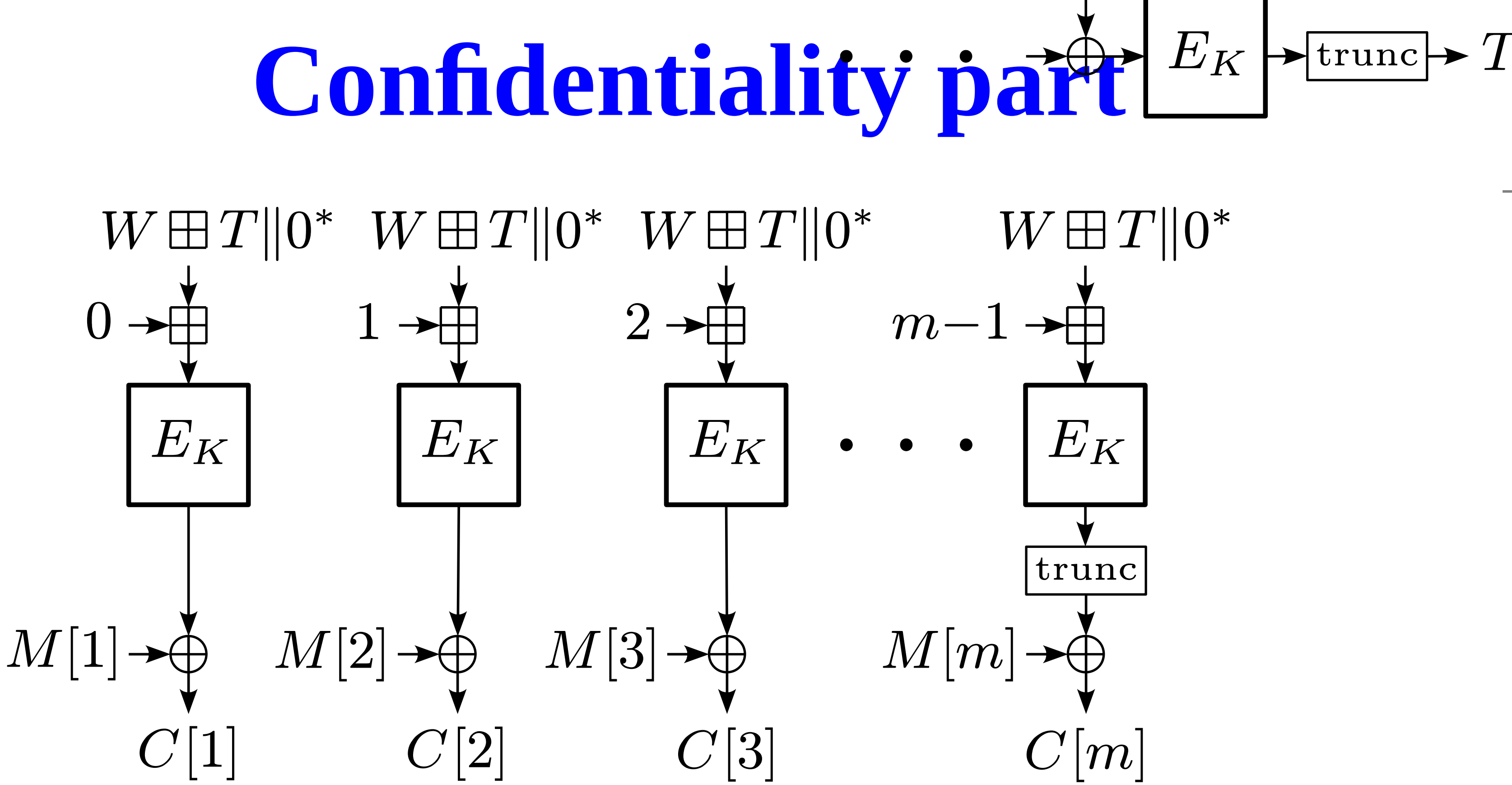
Associated data part



Tag generation



Confidentiality part



DAE security

DAE is "secure" if $(\mathcal{E}_K, \mathcal{D}_K)$ and $(\mathcal{R}, \perp)$ are indistinguishable.

Intuition: DAE is "secure" if the output of encryption looks random (as long as the input is "new"), and the forgery is hard.

Security claim

Our PMAC+CTR is a secure DAE under the assumption that E is a pseudorandom permutation (up to the standard birthday bound security).

A proof is not simple, and we are still developing the details of the proof, but the probability of "bad events" can be shown to be small.

AES-NI

- **AES-NI $\ni$ `aesenc`, `aesenclast`.**

Core	Latency	Throughput
Nehalem	6	2
Sandy Bridge	8	1

Unit: clock cycle

- **AES-128 can be implemented as `pxor` + `aesenc` $\times 9$ + `aesenclast`.**

Core	Sequential	Parallel
Nehalem	3.813	1.271
Sandy Bridge	5.063	0.646

Unit: clock cycle per byte

AES-NI on Sandy Bridge

- **Repeating independent `aesenc`s achieves the throughput of 1 clock cycle.**
- **Adding any instruction even which is zero cycle instruction such as `xor rax, rax` requires more clock cycles.**
- **That is, `aesenc` seems MSROM instruction, which can only be decoded 1 instruction per cycle.**
  It means that it is hard to mix other instructions in the sequence of AES encryption.

Incrementing Gray Code and “Doubling”

Gray code:

$$L \leftarrow L \oplus \texttt{tbl}[\mathrm{ntz}(\mathtt{ctr}++)]$$

Doubling:

$$L \leftarrow (L^{\ll 1}) \oplus (\mathrm{msb}(L) \cdot \mathtt{0x87})$$

Optimizing Doubling

```
1: movmskpd rax, xmm0
2: add      rax, rax
3: paddq    xmm0, xmm0
4: pxor     xmm0, tbl[8 * rax]
5: vpshufb  xmm2, xmm0, xmm1
```

`movmskpd rax, xmm0`:

$(\boldsymbol{h}, \boldsymbol{l}) \leftarrow \texttt{xmm0}$

$\texttt{rax} \leftarrow \mathbf{0} \cdots \mathbf{0} \| \mathbf{msb}(\boldsymbol{h}) \| \mathbf{msb}(\boldsymbol{l})$

$\texttt{tbl} = [\mathbf{0}, \mathbf{1}^{\ll \mathbf{64}}, \texttt{0x87}, \mathbf{1}^{\ll \mathbf{64}} \oplus \texttt{0x87}]$

Performance Comparison on Sandy Bridge

Mode	cycles per byte	Source
CTR	0.916	OpenSSL 1.0.1c
	0.787	Ours
GCM	2.900	OpenSSL 1.0.1c
	2.59	Gueron@IndoC'11
PMAC+CTR	2.047	Ours

Conclusion

- **PMAC+CTR is faster than GCM on Sandy Bridge.**
- **PMAC+CTR is DAE, fully parallelizable, and uses single key, and internal blockcipher is used only for forward direction.**
- **OCB3 seems an efficiency-optimal NAE mode, and considering better DAE modes with AES-NI seems an interesting topic.**

