# Hierarchical Approach in RNS Base Extension for Asymmetric Cryptography

Libey Djath[1], Karim Bigou[1], Arnaud Tisserand[2]

[1] Université de Bretagne Occidentale / Lab-STICC, UMR CNRS 6285
[2] CNRS / Lab-STICC, UMR 6285

ARITH-26, 10-12 June 2019, Kyoto, Japan

# Contents

# Context

Asymmetric cryptography serves in:

- digital signature
- authentication
- secret key exchange

An example of asymmetric cryptosystem:

- Elliptic Curve Cryptography (ECC) [Mil85, Kob87]

For ECC, computations are performed in $GF(P)$ with $P$ a $200 - 500$ bits prime

1 ECC primitive requires a thousand of additions, subtractions and multiplications modulo $P$

# Residue Number System (RNS)

## RNS

- non-positional representation system
- Chinese Remainder Theorem (CRT)
- $X$ is represented by its residues over a base
- representation with internal parallelism

## RNS base

An RNS base $\mathcal{A}$ is a tuple $(a_1, a_2, ..., a_n)$ of coprime integers named moduli

## Representing the number $X$

$$\overrightarrow{X} = (X \bmod a_1 \ , \quad X \bmod a_2 \ , \quad \dots \quad , \quad X \bmod a_n \ )$$

$$\overrightarrow{X} = (\quad x_{a_1} \quad , \quad x_{a_2} \quad , \quad \dots \quad , \quad x_{a_n} \quad )$$

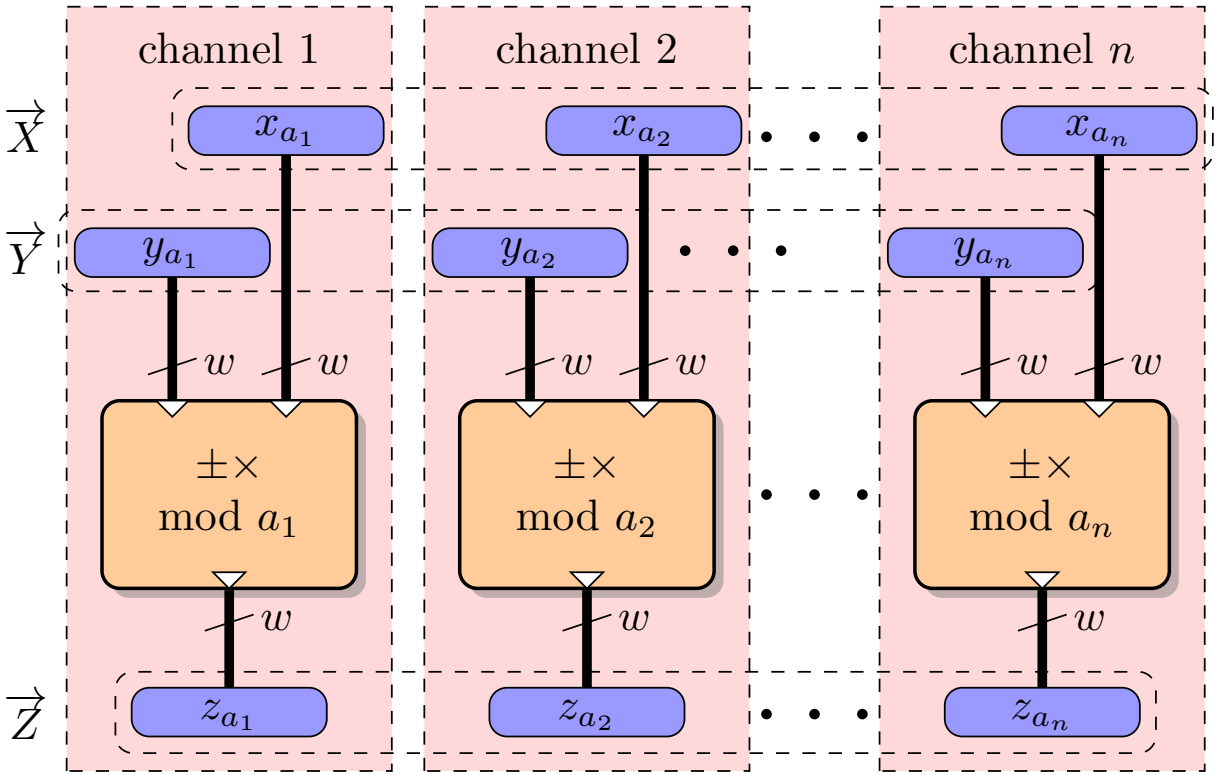
## Converting back to positional representation

Compute the CRT over all the $x_{a_i}$s in base $\mathcal{A}$

# RNS

In hardware implementations of asymmetric cryptosystems:

- large integers are splitted in small residues (typically 16-64 bits integers)
- computations on large integers are replaced by parallel computations on small residues




$a_i$ are pseudo Mersenne for efficiency purpose

# RNS

Main advantages of RNS architectures:

- carry free operations among the channels
- fast parallel $+$, $-$, $\times$
- random order internal computations

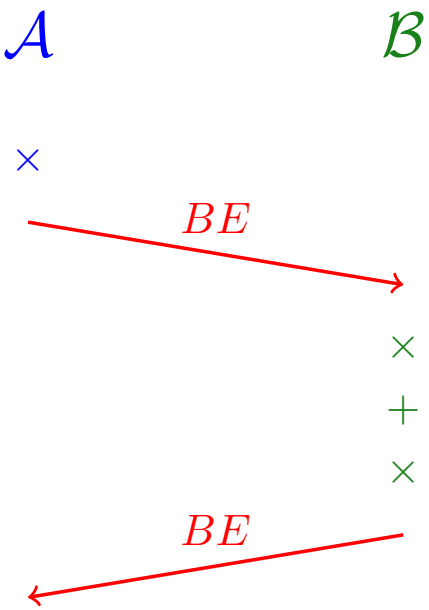
Drawback:

- Comparison, division and mod $P$ reduction are difficult

# RNS Montgomery mod $P$ Reduction [PP95]

**Input:** $X_A,\ X_B$
**Precomp.:** $P_A, P_B,\quad \left(-P^{-1}\right)_A,\quad \left(A^{-1}\right)_B$
**Output:** $S_A$ and $S_B$ with $S = \left(XA^{-1}\right) \bmod P + \delta P$ and $\delta \in \{0, 1, 2\}$

1 $Q_A \leftarrow X_A \times \left(-P^{-1}\right)_A$
2 $Q_B \leftarrow BE\left(Q_A, \mathcal{A}, \mathcal{B}\right)$
3 $R_B \leftarrow X_B + Q_B \times P_B$
4 $S_B \leftarrow R_B \times \left(A^{-1}\right)_B$
5 $S_A \leftarrow BE\left(S_B, \mathcal{B}, \mathcal{A}\right)$
6 **return** $(S_A, S_B)$



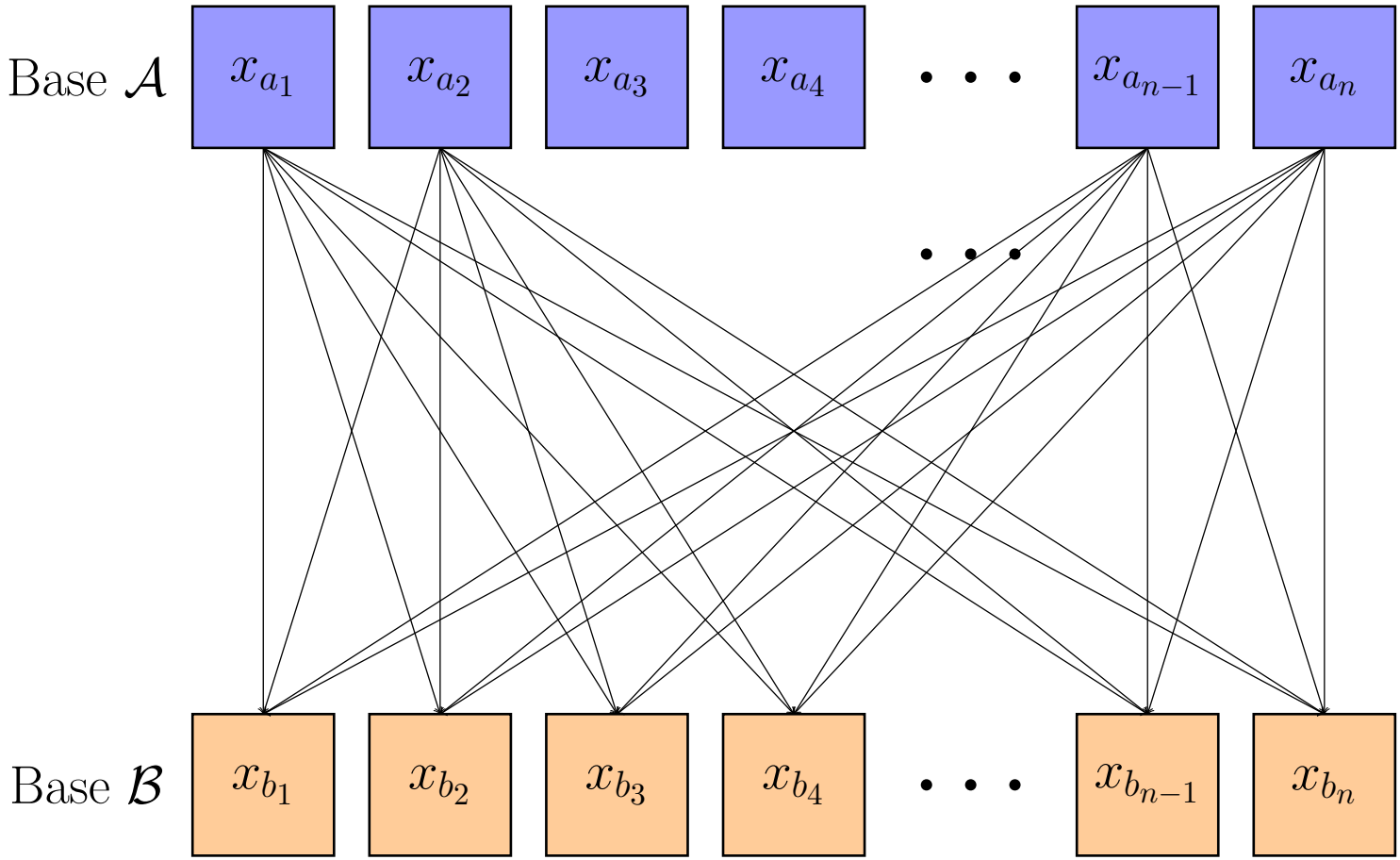

BE: base extension

## Chinese Remainder Theorem (CRT) formula

$$X = \left| \sum_{i=1}^{n} |x_{a_i} \times \left(\frac{A}{a_i}\right)^{-1}|_{a_i} \times \frac{A}{a_i} \right|_A = \left( \sum_{i=1}^{n} |x_{a_i} \times \left(\frac{A}{a_i}\right)^{-1}|_{a_i} \times \frac{A}{a_i} \right) - hA$$

with $A = a_1 \times \ldots \times a_n$

# Base Extension (BE) [KKSS00]




BE converts $X$ in base $\mathcal{A}$ into X in base $\mathcal{B}$

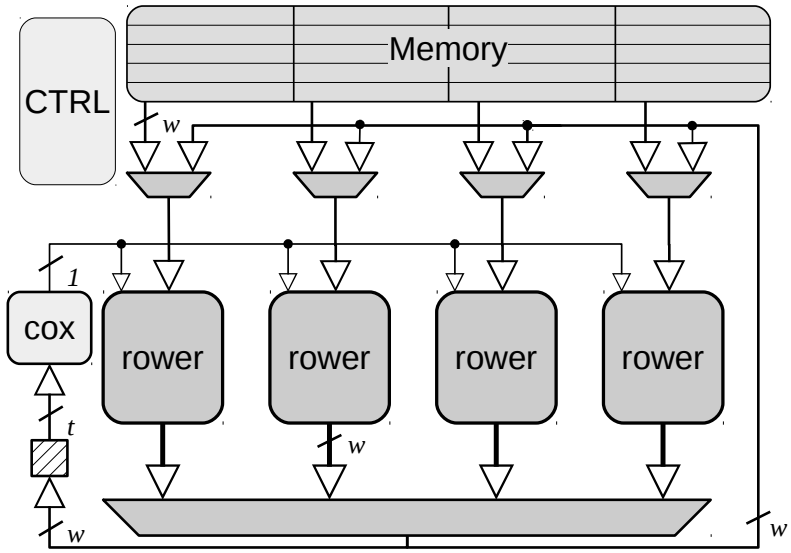
# Base Extension [KKSS00]

## BE algorithm from [KKSS00]

**Input:** $X_A$, $\sigma = 0$ or $0.5$
**Precomp.:** $T_{a_i}\ \forall i \in [1, n]$
**Output:** $X_B$

1 **for** $i$ **from** $1$ **to** $n$ **parallel do**
2 $\quad \widehat{x}_{a_i} \leftarrow |x_{a_i} \times T_{a_i}|_{a_i}$
3 **for** $i$ **from** $1$ **to** $n$ **do**
4 $\quad \sigma \leftarrow \sigma + \frac{\mathrm{trunc}(\widehat{x}_{a_i})}{2^w}$
5 $\quad h_i \leftarrow \lfloor \sigma \rceil$
6 $\quad \sigma \leftarrow \sigma - h_i$
7 $\quad$ **for** $k$ **from** $1$ **to** $n$ **parallel do**
8 $\quad\quad x_{b_k} \leftarrow \left| x_{b_k} + \widehat{x}_{a_i} \times \left|\frac{A}{a_i}\right|_{b_k} + |-h_i\, A|_{b_k} \right|_{b_k}$

## Cox-rower architecture from [Gui10]




State of the art solution is usually called KBE

# Contents

# Idea of Hierarchical Base Extension (HBE)

## Changing the notation

$$\mathcal{A} = (a_1 \quad \cdots \quad a_n) \qquad \mathcal{A} = \begin{pmatrix} a_{1,1} & \cdots & a_{1,c} \\ \vdots & \cdots & \vdots \\ a_{r,1} & \cdots & a_{r,c} \end{pmatrix}$$

with $n = r \times c$

## Main Idea

- gather residues by row ($c$ residues per row) into super-residues in base $\mathcal{A}$ by computing their partial CRTs
- compute the CRT of the super-residues of base $\mathcal{A}$ in base $\mathcal{B}$

# Rewriting the KBE Algorithm

## 1D KBE

**Input:** $X_A$, $\sigma = 0$ or $0.5$
**Precomp.:** $T_{a_i}\ \forall i \in [1, n]$
**Output:** $X_B$

1 **for** $i$ **from** 1 **to** $n$ **parallel do**
2 $\quad \widehat{x}_{a_i} \leftarrow |x_{a_i} \times T_{a_i}|_{a_i}$
3 **for** $i$ **from** 1 **to** $n$ **do**
4 $\quad \sigma \leftarrow \sigma + \frac{\mathrm{trunc}(\widehat{x}_{a_i})}{2^w}$
5 $\quad h_i \leftarrow \lfloor \sigma \rfloor$
6 $\quad \sigma \leftarrow \sigma - h_i$
7 $\quad$ **for** $k$ **from** 1 **to** $n$ **parallel do**
8 $\quad\quad x_{b_k} \leftarrow \left| x_{b_k} + \widehat{x}_{a_i} \times \left|\frac{A}{a_i}\right|_{b_k} + |-h_i\, A|_{b_k} \right|_{b_k}$

Main cost: $n^2$ executions of line 8

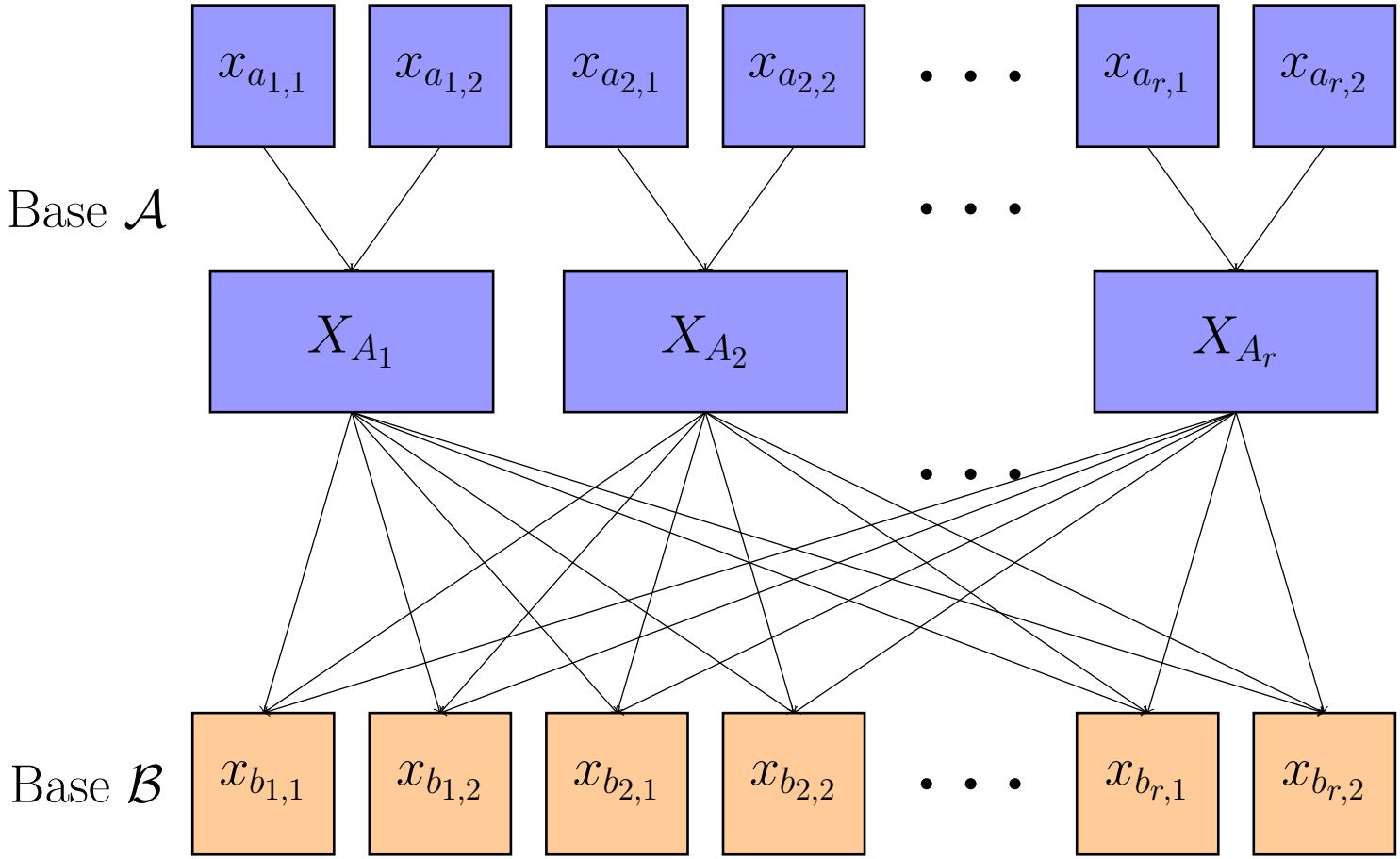
## 2D KBE

**Input:** $X_A$, $\sigma = 0$ or $0.5$
**Precomp.:** $T_{a_{i,j}}\ \forall i \in [1, r]$ and $\forall j \in [1, c]$
**Output:** $X_B$

1 **for** $i$ **from** 1 **to** $r$ **parallel do**
2 $\quad$ **for** $j$ **from** 1 **to** $c$ **parallel do**
3 $\quad\quad \widehat{x}_{a_{i,j}} \leftarrow |x_{a_{i,j}} \times T_{a_{i,j}}|_{a_{i,j}}$
4 **for** $i$ **from** 1 **to** $r$ **do**
5 $\quad$ **for** $j$ **from** 1 **to** $c$ **do**
6 $\quad\quad \sigma \leftarrow \sigma + \frac{\mathrm{trunc}(\widehat{x}_{a_{i,j}})}{2^w}$
7 $\quad\quad h_{i,j} \leftarrow \lfloor \sigma \rfloor$
8 $\quad\quad \sigma \leftarrow \sigma - h_{i,j}$
9 $\quad\quad$ **for** $k$ **from** 1 **to** $r$ **parallel do**
10 $\quad\quad\quad$ **for** $l$ **from** 1 **to** $c$ **parallel do**
11 $\quad\quad\quad\quad x_{b_{k,l}} \leftarrow \left| x_{b_{k,l}} + \widehat{x}_{a_{i,j}} \times \left|\frac{A}{a_{i,j}}\right|_{b_{k,l}} + |-h_{i,j}\, A|_{b_{k,l}} \right|_{b_{k,l}}$

With $n = r \times c$, main cost:
$r^2 c^2$ executions of line 11

# HBE ($c = 2$)


Base $\mathcal{A}$
$x_{a_{1,1}}$
$x_{a_{1,2}}$
$x_{a_{2,1}}$
$x_{a_{2,2}}$
$x_{a_{r,1}}$
$x_{a_{r,2}}$
$X_{A_1}$
$X_{A_2}$
$X_{A_r}$
Base $\mathcal{B}$
$x_{b_{1,1}}$
$x_{b_{1,2}}$
$x_{b_{2,1}}$
$x_{b_{2,2}}$
$x_{b_{r,1}}$
$x_{b_{r,2}}$

# Comparison between KBE and HBE

## KBE

**Input:** $X_A$, $\sigma = 0$ or $0.5$
**Precomp.:** $T_{a_{i,j}}$ $\forall i \in [1, r]$ and $\forall j \in [1, c]$
**Output:** $X_B$

1 **for** $i$ **from** 1 **to** $r$ **parallel do**
2  **for** $j$ **from** 1 **to** $c$ **parallel do**
3  $\widehat{x}_{a_{i,j}} \leftarrow \left| x_{a_{i,j}} \times T_{a_{i,j}} \right|_{a_{i,j}}$
4 **for** $i$ **from** 1 **to** $r$ **do**
5  **for** $j$ **from** 1 **to** $c$ **do**
6  $\sigma \leftarrow \sigma + \frac{\text{trunc}(\widehat{x}_{a_{i,j}})}{2^w}$
7  $h_{i,j} \leftarrow \lfloor \sigma \rfloor$
8  $\sigma \leftarrow \sigma - h_{i,j}$
9  **for** $k$ **from** 1 **to** $r$ **parallel do**
10  **for** $l$ **from** 1 **to** $c$ **parallel do**
11  $x_{b_{k,l}} \leftarrow$
$\left| x_{b_{k,l}} + \widehat{x}_{a_{i,j}} \times \left| \frac{A}{a_{i,j}} \right|_{b_{k,l}} + | - h_{i,j}\, A|_{b_{k,l}} \right|_{b_{k,l}}$

Main cost: $r^2c^2$ executions of line 11

## HBE

**Input:** $X_A$, $\sigma = 0$ or $0.5$
**Precomp.:** $T_{a_{i,j}}$ $\forall i \in [1, r]$ and $\forall j \in [1, c]$
**Output:** $X_B$

1 **for** $i$ **from** 1 **to** $r$ **parallel do**
2  **for** $j$ **from** 1 **to** $c$ **parallel do**
3  $\widehat{x}_{a_{i,j}} \leftarrow \left| x_{a_{i,j}} \times T_{a_{i,j}} \right|_{a_{i,j}}$
4 **for** $i$ **from** 1 **to** $r$ **parallel do**
5  $\widehat{X}_{A_i} \leftarrow 0$
6  **for** $j$ **from** 1 **to** $c$ **do**
7  $\widehat{X}_{A_i} \leftarrow \widehat{X}_{A_i} + \widehat{x}_{a_{i,j}} \times \overline{a_{i,j}}$ *(no reduction)*
8 **for** $i$ **from** 1 **to** $r$ **do**
9  $\sigma \leftarrow \sigma + \frac{\text{trunc}(\widehat{X}_{A_i})}{2^{w \times c}}$
10  $h_i \leftarrow \lfloor \sigma \rfloor$
11  $\sigma \leftarrow \sigma - h_i$
12  **for** $k$ **from** 1 **to** $r$ **parallel do**
13  **for** $l$ **from** 1 **to** $c$ **parallel do**
14  $\widehat{x}_{b_{k,l,i}} \leftarrow \left| \widehat{X}_{A_i} \right|_{b_{k,l}}$
15  $x_{b_{k,l}} \leftarrow$
$\left| x_{b_{k,l}} + \widehat{x}_{b_{k,l,i}} \times \overline{A_i} + | - h_i A|_{b_{k,l}} \right|_{b_{k,l}}$

Main cost: $r^2c$ executions of line 15

# Theoretical Cost Comparison for $c = 2$

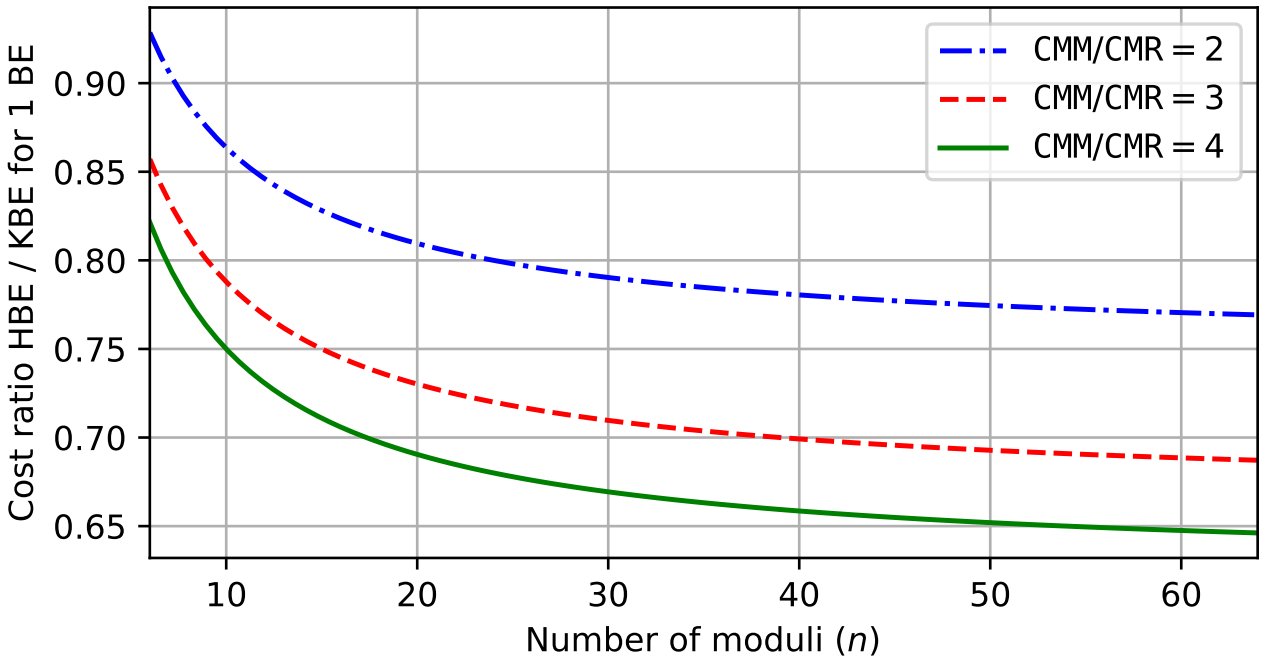
Notation:

- $CMM(w, w)$ for a $(w \times w \bmod w)$-bit modular multiplication
- $CMR(w', w)$ for a $(w' \bmod w)$-bit modular reduction

KBE cost: $n^2\, CMM(w, w) + n\, CMM(w, w)$

HBE cost: $\frac{n^2}{2}\, CMM(w, w) + \frac{n^2}{2}\, CMR(2w + 1, w) + 2n\, CMM(w, w)$

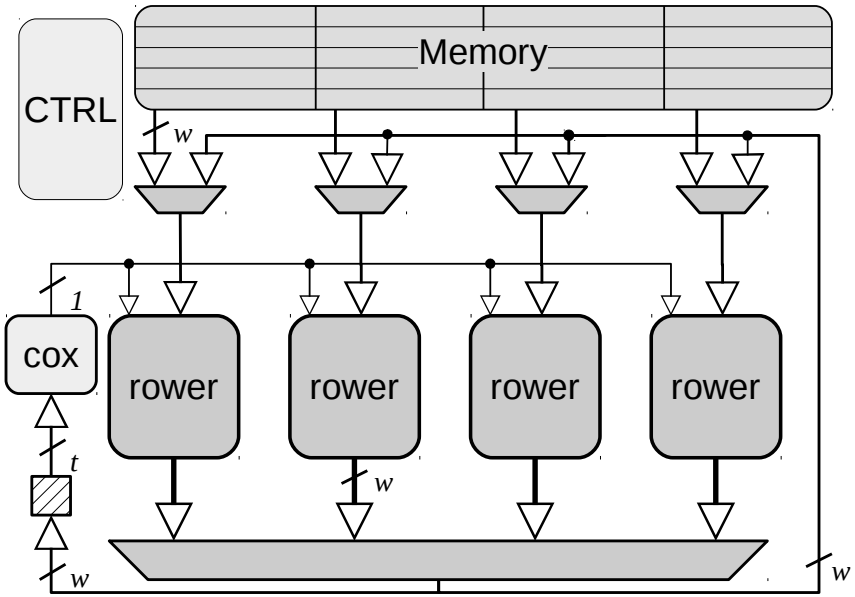
Theoretical cost ratio for one BE for various base sizes (n)



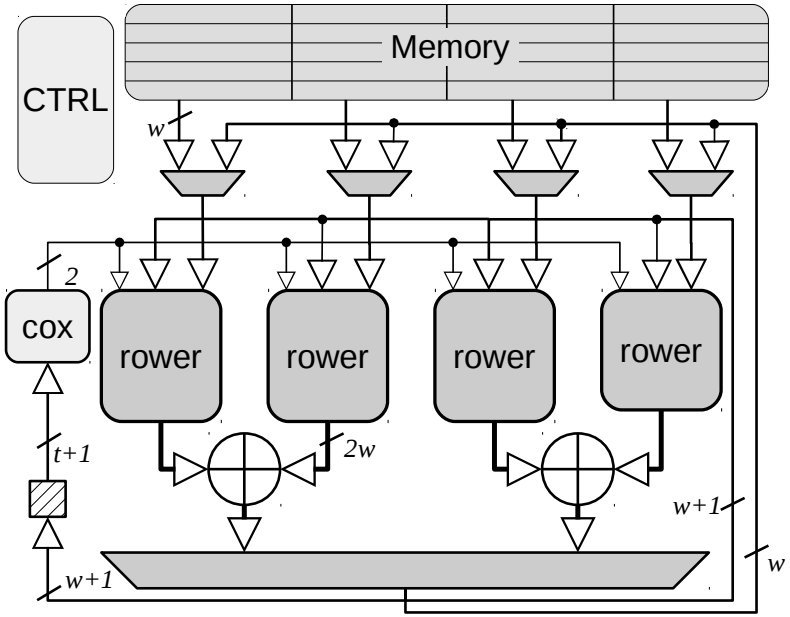
# Contents

# Architecture Descriptions

Cox-rower architecture for KBE [Gui10]




Proposed architecture for HBE ($c = 2$)



# Hardware Implementation

## Target FPGA

ZYNQ-7 ZC702 from Xilinx (ZedBoard xc7z020clg484-1)

## Tool

Vivado HLS (version 2017.4) from Xilinx

## Implementation

- $P$ size $= 256, 384$ bits
- $w = 17, 20, 24, 28$ bits

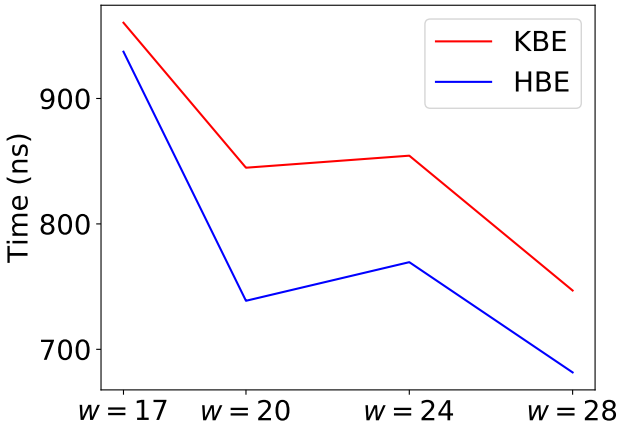
## Optimization

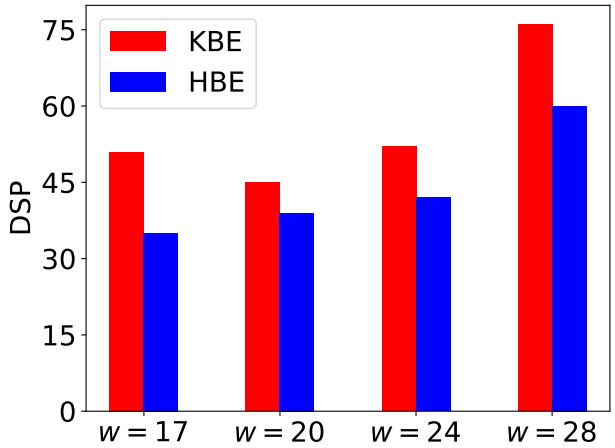
Both algorithms, KBE and HBE ($\underline{c = 2}$) are implemented:

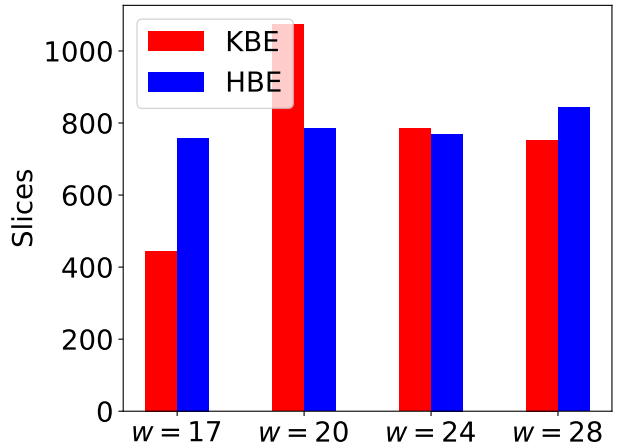
- same manner
- same optimization effort

# Hardware Implementation Results

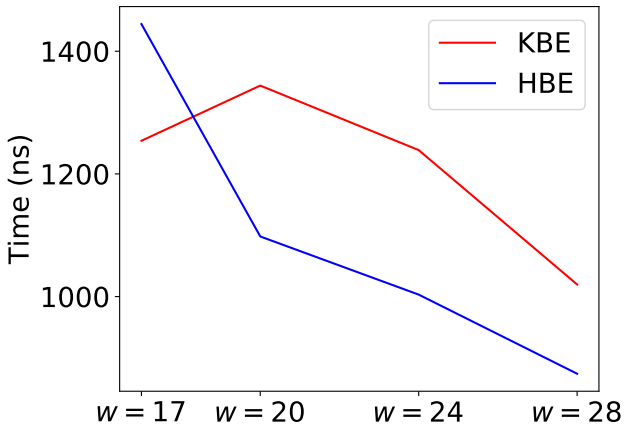
256-bit $P$:

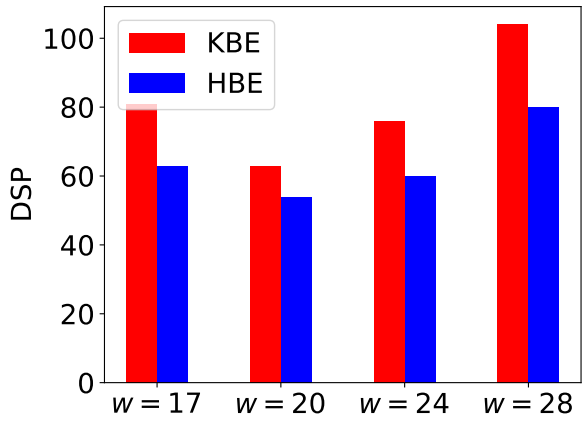


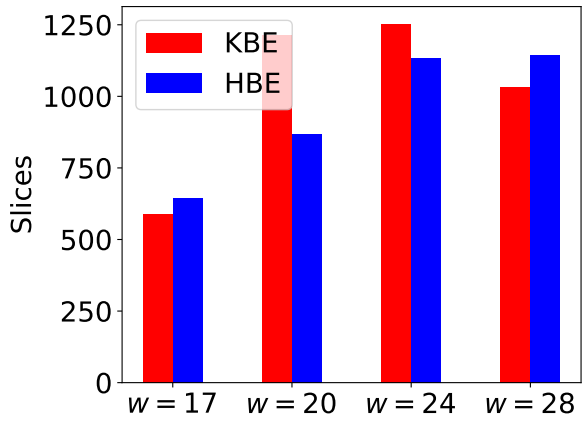




384-bit $P$:




- most of the time, we have a faster AND smaller solution
- no impact on BRAMs and periods

# Contents

# Conclusion

## Conclusion

The proposed hierarchical approach BE:

- improves the main cost of the BE algorithm from $r^2\,c^2$ to $r^2\,c$
- preserves quite well the internal parallelism (for $c = 2$)
- on a XC7Z020 FPGA, it shows an improvement up to 18% in total time and up to 31% in DSPs

## Future Work

- study the architecture for the cases $c = 3, 4$
- implement a full ECC crypto-processor

# References I


[Gui10] N. Guillermin.
A high speed coprocessor for elliptic curve scalar multiplications over $\mathbb{F}_p$.
In *Proc. Cryptographic Hardware and Embedded Systems (CHES)*, volume 6225 of *LNCS*, pages 48–64. Springer, August 2010.

[KKSS00] S. Kawamura, M. Koike, F. Sano, and A. Shimbo.
Cox-Rower architecture for fast parallel Montgomery multiplication.
In *Proc. Internat. Conf. Theory and Application of Cryptographic Techniques (EUROCRYPT)*, volume 1807 of *LNCS*, pages 523–538. Springer, May 2000.

[Kob87] N. Koblitz.
Elliptic curve cryptosystems.
volume 48, pages 203–209. American Mathematical Society, 1987.

[Mil85] V .S. Miller.
Use of elliptic curve in cryptography.
In *Advances in Cryptology*, volume 218, pages 417–426. Springer, 1985.

[PP95] K. C. Posch and R. Posch.
Modulo reduction in residue number systems.
*IEEE Transactions on Parallel and Distributed Systems*, 6(5):449–454, May 1995.

# Hardware Implementation Results

| $\mathbb{F}_P$ width (bits) | BE algo. | KBE | HBE | KBE | HBE | KBE | HBE | KBE | HBE |
|---|---|---|---|---|---|---|---|---|---|
| | $w$ (bits) | 17 | | 20 | | 24 | | 28 | |
| 256 | nb. slices | 445 | 758 | 1073 | 784 | 785 | 769 | 753 | 843 |
| | nb. DSP | 51 | 35 | 45 | 39 | 52 | 42 | 76 | 60 |
| | nb. BRAM | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | period (ns) | 9.8 | 10.3 | 9.6 | 8.9 | 9.6 | 9.5 | 9.7 | 9.6 |
| | nb. cycles | 98 | 91 | 88 | 83 | 89 | 81 | 77 | 71 |
| | time (ns) | 960.4 | 937.3 | 844.8 | 738.7 | 854.4 | 769.5 | 746.9 | 681.6 |
| 384 | nb. slices | 587 | 644 | 1215 | 869 | 1251 | 1134 | 1031 | 1145 |
| | nb. DSP | 81 | 63 | 63 | 54 | 76 | 60 | 104 | 80 |
| | nb. BRAM | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | period (ns) | 7.6 | 10.1 | 9.6 | 9.0 | 7.6 | 7.6 | 9.9 | 9.4 |
| | nb. cycles | 165 | 143 | 140 | 122 | 163 | 132 | 103 | 93 |
| | time (ns) | 1254.0 | 1444.3 | 1344.0 | 1098.0 | 1238.8 | 1003.2 | 1019.7 | 874.2 |

HLS implementation results on a XC7Z020 FPGA for our HBE and the KBE (from [KKSS00]) algorithms for two widths of prime field elements and four RNS channel widths $w$.