RNS Modular Multiplication through Reduced Base Extensions

Karim Bigou and Arnaud Tisserand

INRIA-IRISA-CAIRN

ASAP Conference June 18-20

Context and Objectives

Research group main objective:

Design hardware cryptoprocessors for asymmetric cryptography on FPGA and ASIC with advanced arithmetic support

Various aspects of arithmetic operators:

- efficient algorithms
- fast and protected representations of numbers
- hardware implementations

This work:

Faster Modular multiplication for cryptographic computations in the residue number system (RNS)

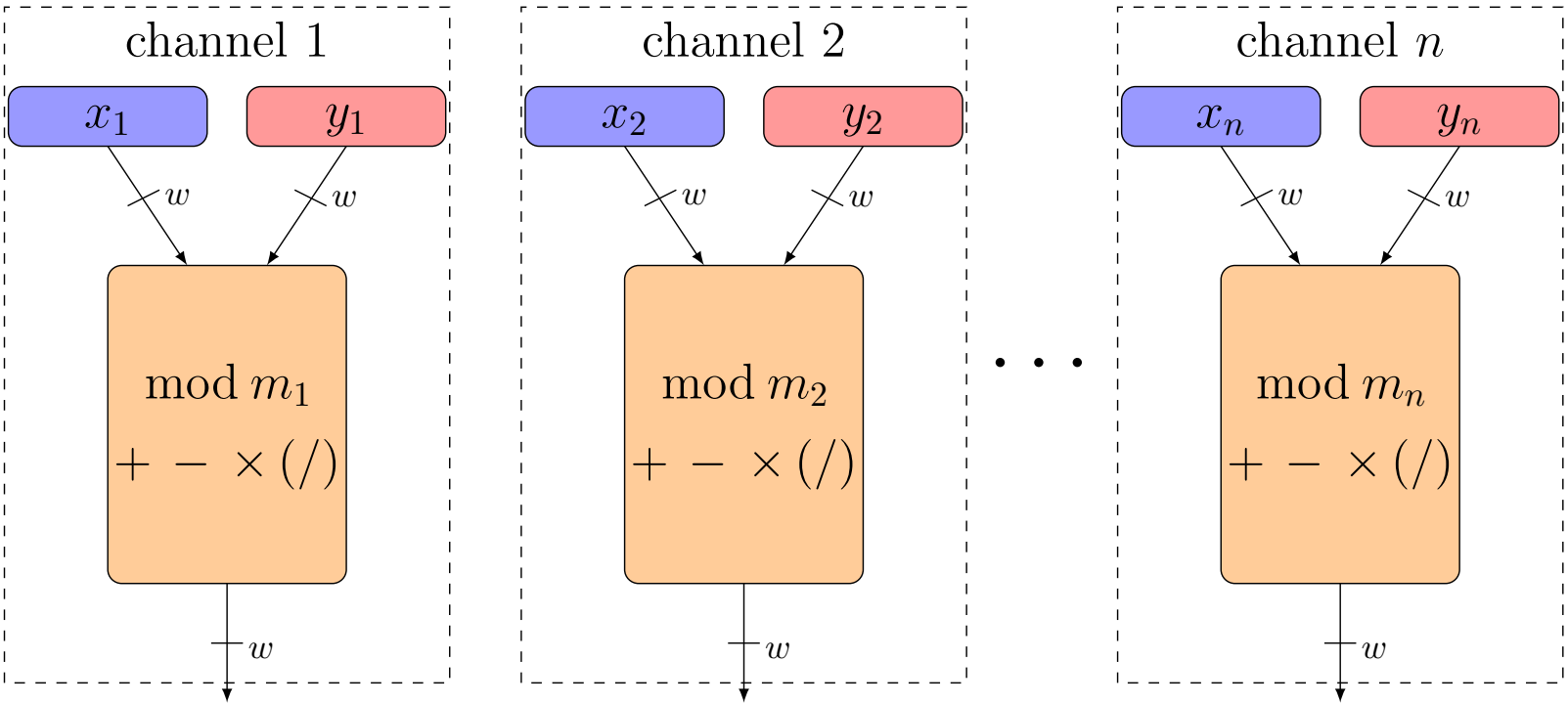
Residue Number System (RNS) [5] [3]

X and Y two large integers (from 160 to 4096 bits) are represented by:

$\overrightarrow{X} = (x_1, \ldots, x_n) = (X \bmod m_1, \ldots, X \bmod m_n)$

$\overrightarrow{Y} = (y_1, \ldots, y_n) = (Y \bmod m_1, \ldots, Y \bmod m_n)$

Modular operations over w-bit chunks, e.g. w is 16–64



RNS base $\mathcal{B} = (m_1, \ldots, m_n)$, n pairwise co-prime integers of w bits with $n \times w \geqslant \log_2 P$

RNS Properties

Pros:

- **Carry-free** between channels
  - each channel is independant
- **Fast parallel** $+$, $-$, $\times$ and some exact divisions
  - computations over all channels can be performed in parallel
  - a multiplication requires n elementary modular multiplications (`EMM`)
- **Non-positional** number system
  - randomization of computations (SCA countermeasures)

Cons:

- comparison, **modular reduction** (by P prime) and division are hard

RNS Base Extension [6]

- Usual technique for modular reduction: add redundancy using 2 bases
- $\mathcal{B}_a = (m_{a,1}, \ldots, m_{a,n})$ and $\mathcal{B}_b = (m_{b,1}, \ldots, m_{b,n})$ are coprime RNS bases
- X is $\overrightarrow{X_a}$ in $\mathcal{B}_a$ and $\overrightarrow{X_b}$ in $\mathcal{B}_b$
- The base extension (BE, introduced in [6]) is defined by:

$$\overrightarrow{X_b} = BE(\overrightarrow{X_a}, \mathcal{B}_a, \mathcal{B}_b)$$

- Some operations become possible after a base extension
  - $M_a = \prod_{i=1}^{n} m_{a,i}$ is invertible in $\mathcal{B}_b$
  - exact division by M_a can be done easily
- State-of-art BE algorithms cost $n^2 + n$ w-bit `EMM`s

RNS Montgomery Reduction (RNS-MR) [4, 1]

Input: $\overrightarrow{X_a}$, $\overrightarrow{X_b}$ with $X < \alpha P^2 < PM$ and $2P < M'$

Output: $\overrightarrow{\omega}_{a|b}$ with $\omega \equiv X \times M^{-1} \bmod P$ $0 \leqslant \omega < 2P$

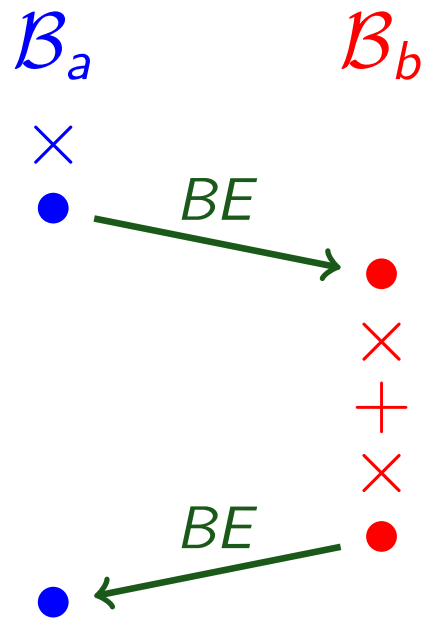
$\overrightarrow{Q}_a \longleftarrow \overrightarrow{X_a} \times \overrightarrow{(-P^{-1})}_a$ (in base $\mathcal{B}$)

$\overrightarrow{Q_b} \longleftarrow BE(\overrightarrow{Q_a}, \mathcal{B}_a, \mathcal{B}_b)$

$\overrightarrow{S_b} \longleftarrow \overrightarrow{X_b} + \overrightarrow{Q_b} \times \overrightarrow{P_b}$ (in base $\mathcal{B}_b$)

$\overrightarrow{\omega_b} \longleftarrow \overrightarrow{S_b} \times \overrightarrow{M_a^{-1}}$ (in base $\mathcal{B}_b$)

$\overrightarrow{\omega_a} \longleftarrow BE(\overrightarrow{\omega_b}, \mathcal{B}_b, \mathcal{B}_a)$



RNSMR cost: $2\,n^2 + O(n)$ EMMs

How to exploit RNS properties?

Maximize the use of fully parallelizable operations, e.g. computing patterns in the form of $(AB + CD) \bmod P$

Proposed Modular Multiplication

Idea:

- Split operands into 2 parts and introduce sub-reductions
- only $\frac{3}{2}n$ moduli required vs $2n$ (3 bases of $n/2$)

Constraint:

- Requires an hypothesis on P: not possible for RSA but possible for ECC and discrete logarithm

Operations	$AB \bmod P$	$A^2 \bmod P$	$Cst \times A \bmod P$
MM [EMM]	$2n^2 + 4n$	$2n^2 + 4n$	$2n^2 + 4n$
SPRR [EMM]	$2.5n^2 + 12.5n$	$1.75n^2 + 10.5n$	$1.75n^2 + 7n$

Note: Karatsuba-Offman idea does not work in RNS

Proposed Modular Multiplication Algorithm

Input: $X, Y < \alpha P$
Precomp.: $D = |M_a^{-1}|_P$
Output: $\overrightarrow{V_{a|b|c}}$ with $V \equiv \left|XYM_a^{-1}M_b^{-1}\right|_P$ and $V < \alpha P$
begin

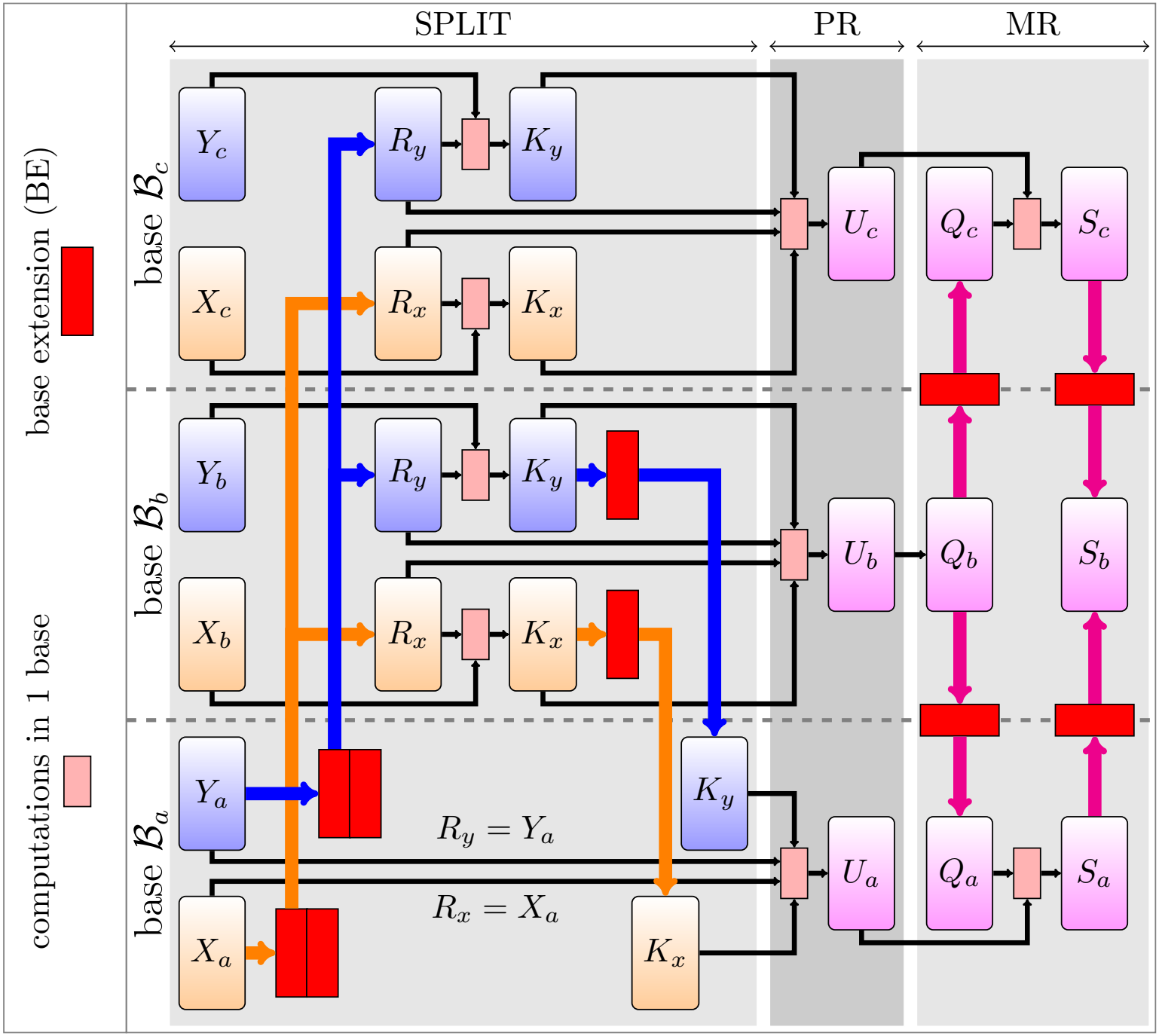
$(\overrightarrow{(K_x)_{a|b|c}}, \overrightarrow{(R_x)_{a|b|c}}) \leftarrow \text{Split}(\overrightarrow{X_{a,b,c}})$

$(\overrightarrow{(K_y)_{a|b|c}}, \overrightarrow{(R_y)_{a|b|c}}) \leftarrow \text{Split}(\overrightarrow{Y_{a,b,c}})$

$\overrightarrow{U_{a|b|c}} \leftarrow \text{PR}\left(\overrightarrow{(K_x)_{a|b|c}}, \overrightarrow{(R_x)_{a|b|c}}, \overrightarrow{(K_y)_{a|b|c}}, \overrightarrow{(R_y)_{a|b|c}}, D\right)$

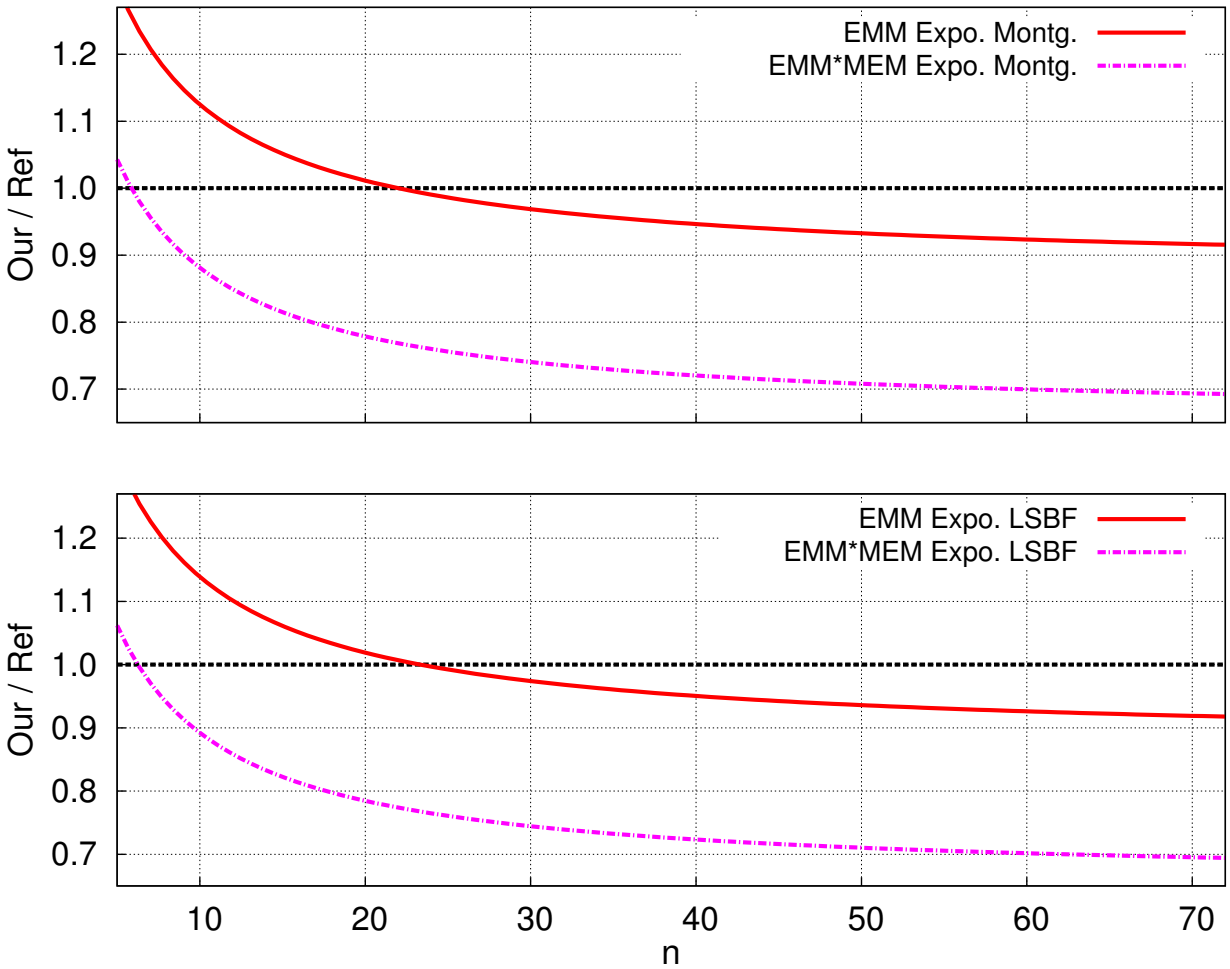
$\overrightarrow{V_{a|b|c}} \leftarrow \text{RNS-MR}(\overrightarrow{U_b}, \overrightarrow{U_{a|c}})$

return $\overrightarrow{V_{a|b|c}}$


SPLIT
PR
MR
base extension (BE)
computations in 1 base
base $\mathcal{B}_c$
base $\mathcal{B}_b$
base $\mathcal{B}_a$
Y_c
X_c
R_y
R_x
K_y
K_x
U_c
Q_c
S_c
Y_b
X_b
U_b
Q_b
S_b
Y_a
X_a
$R_y = Y_a$
$R_x = X_a$
U_a
Q_a
S_a

Theoretical Performance Comparison

Results for exponentiation for discrete logarithm (Diffie-Hellman or ElGamal protocols)



State-of-art reference (Ref):[2]

Conclusion

Our proposition:

- reduces by 25 % the number of precomputations stored
- reduces the number of EMMs up to 10 % for large cryptographic parameters
- reduces by 25 % the number of base elements required

Future works on hardware implementation:

- implementation of the new RNS modular multiplication in full cryptosystems
- time×area trade-off explorations

Thank you for your attention

This work has been supported in part by a PhD grant from *DGA–INRIA* and by the PAVOIS project (ANR 12 BS02 002 01).

References

[1] J.-C. Bajard, L.-S. Didier, and P. Kornerup.
An RNS montgomery modular multiplication algorithm.
IEEE Transactions on Computers, 47(7):766–776, July 1998.

[2] F. Gandino, F. Lamberti, G. Paravati, J.-C. Bajard, and P. Montuschi.
An algorithmic and architectural study on montgomery exponentiation in RNS.
IEEE Transactions on Computers, 61(8):1071–1083, August 2012.

[3] H. L. Garner.
The residue number system.
IRE Transactions on Electronic Computers, EC-8(2):140–147, June 1959.

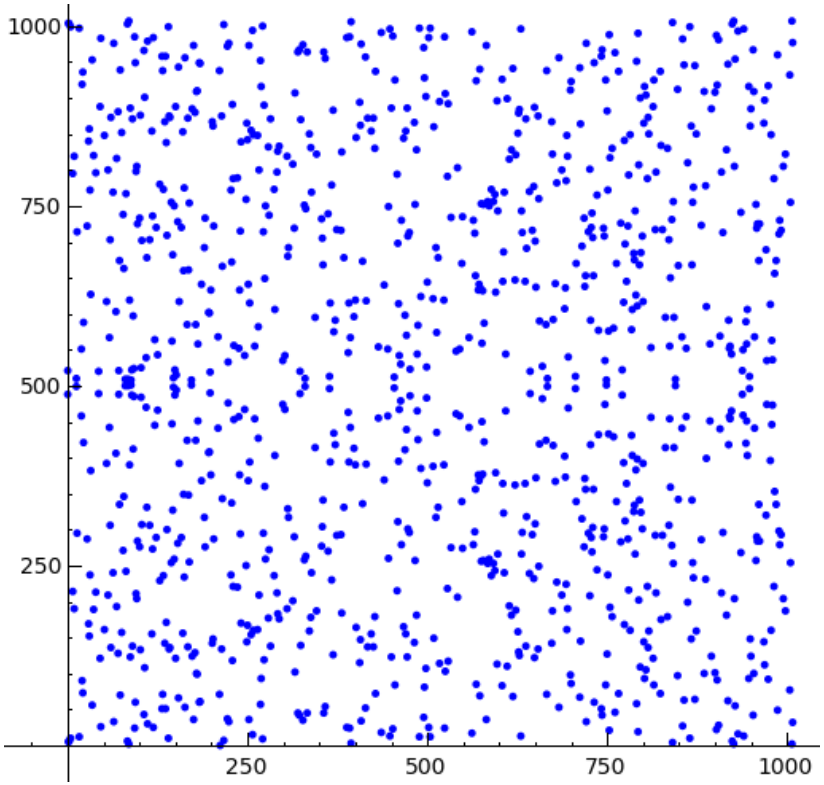
[4] K. C. Posch and R. Posch.
Modulo reduction in residue number systems.
IEEE Transactions on Parallel and Distributed Systems, 6(5):449–454, May 1995.

[5] A. Svoboda and M. Valach.
Operátorové obvody (operator circuits in czech).
Stroje na Zpracování Informací (Information Processing Machines), 3:247–296, 1955.

[6] N. S. Szabo and R. I. Tanaka.
Residue arithmetic and its applications to computer technology.
McGraw-Hill, 1967.

Elliptic Curve Cryptography (ECC)

P large prime of 160–600 bits



$y^2 = x^3 + 4x + 20$ over $\mathbb{F}_{1009}$

Elliptic curve E over $\mathbb{F}_P$:

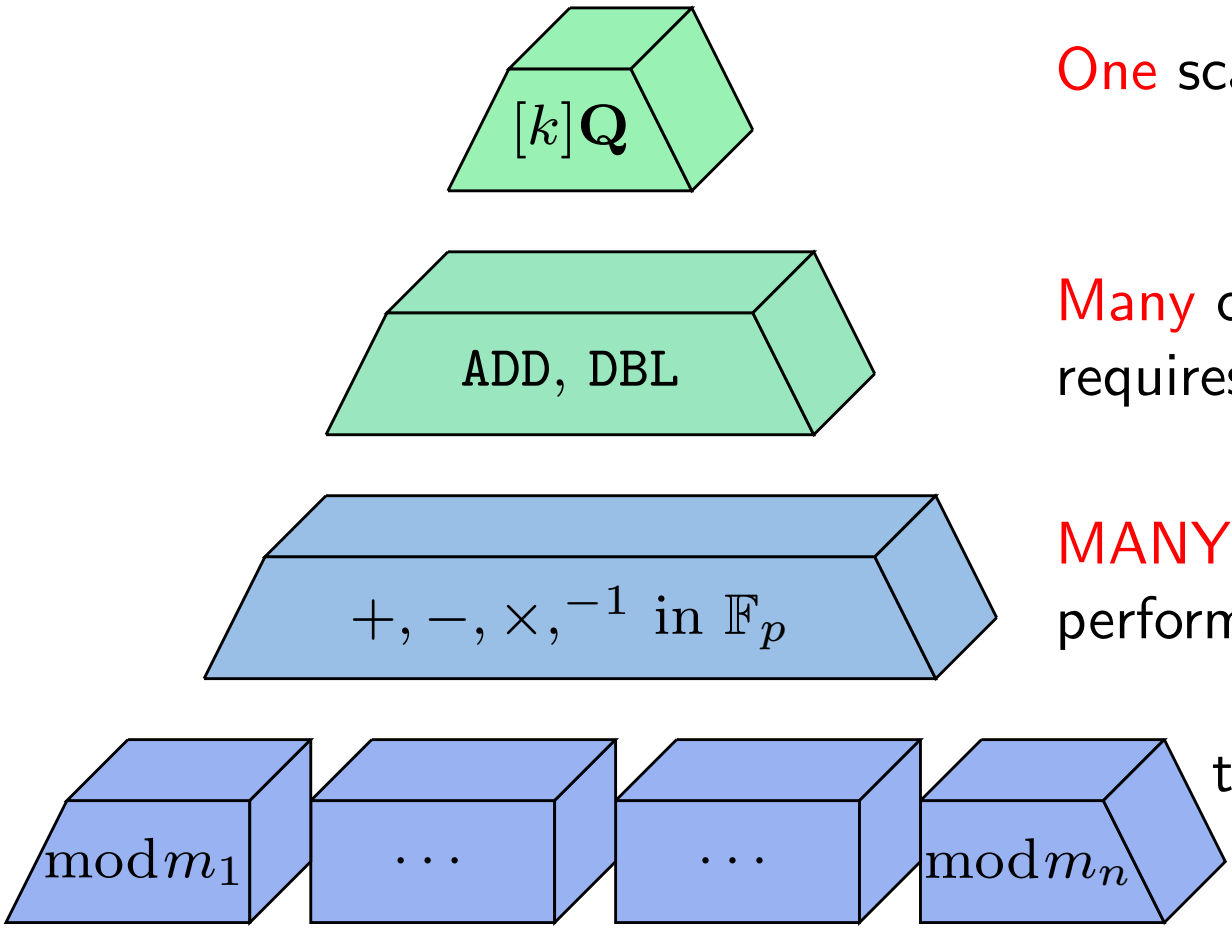
$$y^2 = x^3 + a\,x + b$$

Curve level operations:

- Point addition (`ADD`): $\mathbf{Q} + \mathbf{Q'}$
- Point doubling (`DBL`): $\mathbf{Q} + \mathbf{Q}$
- Scalar multiplication:
  $[k]\mathbf{Q} = \underbrace{\mathbf{Q} + \mathbf{Q} + \ldots + \mathbf{Q}}_{k \text{ times}}$

Security (ECDLP): knowing $\mathbf{Q}$ and $[k]\mathbf{Q}$, k cannot be recovered

Scalar Multiplication Internal Operations



One scalar multiplication requires...

Many curve level operations which requires...

MANY $\mathbb{F}_p$ operations which can be performed using...

the residue number system (RNS)

Ratio SPRR/RNS-MR for ECC Operations