Arithmetic Operators for Pairing-Based Cryptography

Jean-Luc Beuchat

Laboratory of Cryptography and Information Security
Graduate School of Systems and Information Engineering
University of Tsukuba
1-1-1 Tennodai, Tsukuba
Ibaraki, 305-8573, Japan
mailto:beuchat@risk.tsukuba.ac.jp

Joint work with Nicolas Brisebarre (Université J. Monnet, Saint-Étienne, France), Jérémie Detrey (ENS Lyon, France), Eiji Okamoto (University of Tsukuba, Japan), Masaaki Shirase (Future University, Hakodate, Japan), and Tsuyoshi Takagi (Future University, Hakodate, Japan)

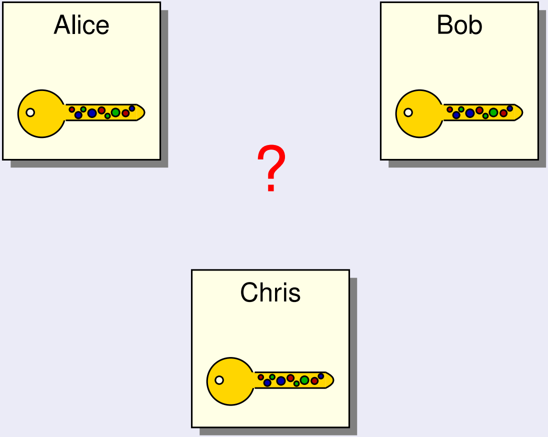
Outline of the Talk

1. Example: Three-Party Key Agreement
2. Computation of the η_T Pairing
3. A Coprocessor for the Full Pairing Computation
4. Conclusion

Example: Three-Party Key Agreement

Key agreement

How can Alice, Bob, and Chris agree upon a shared secret key?



Example: Three-Party Key Agreement

Discrete logarithm problem (DLP)

- $G = \langle P \rangle$: additively-written group of order n
- DLP: given P, Q, find the integer $x \in \{0, \ldots, n-1\}$ such that $Q = xP$

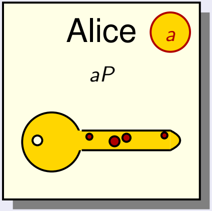
Example: Three-Party Key Agreement

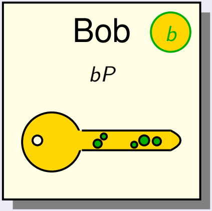
Discrete logarithm problem (DLP)

- $G = \langle P \rangle$: additively-written group of order n
- DLP: given P, Q, find the integer $x \in \{0, \ldots, n-1\}$ such that $Q = xP$

Diffie-Hellman problem (DHP)

Given P, aP, and bP, find abP.





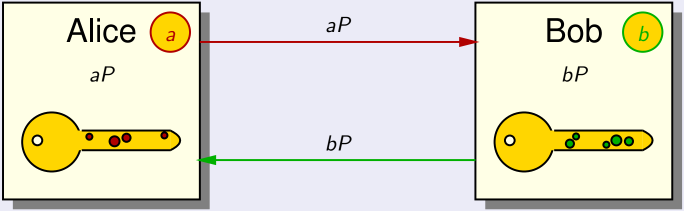
Example: Three-Party Key Agreement

Discrete logarithm problem (DLP)

- $G = \langle P \rangle$: additively-written group of order n
- DLP: given P, Q, find the integer $x \in \{0, \ldots, n-1\}$ such that $Q = xP$

Diffie-Hellman problem (DHP)

Given P, aP, and bP, find abP.



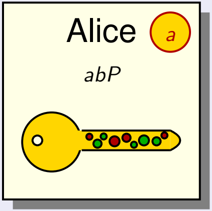
Example: Three-Party Key Agreement

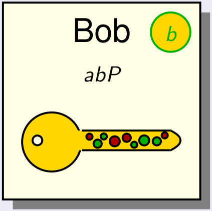
Discrete logarithm problem (DLP)

- $G = \langle P \rangle$: additively-written group of order n
- DLP: given P, Q, find the integer $x \in \{0, \ldots, n-1\}$ such that $Q = xP$

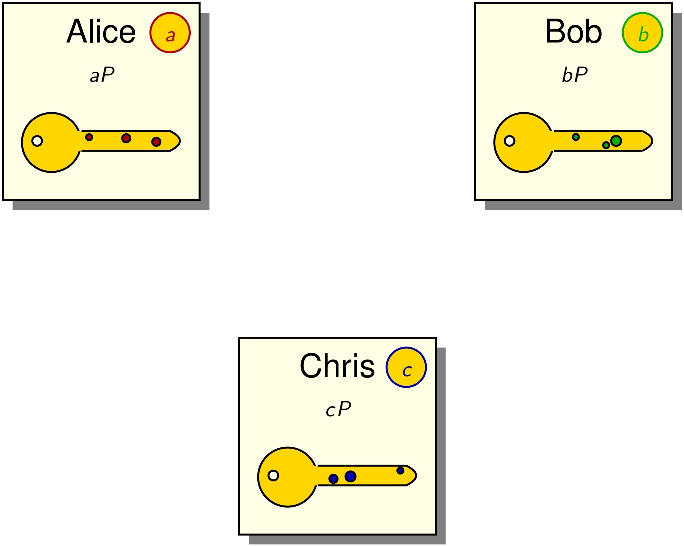
Diffie-Hellman problem (DHP)

Given P, aP, and bP, find abP.

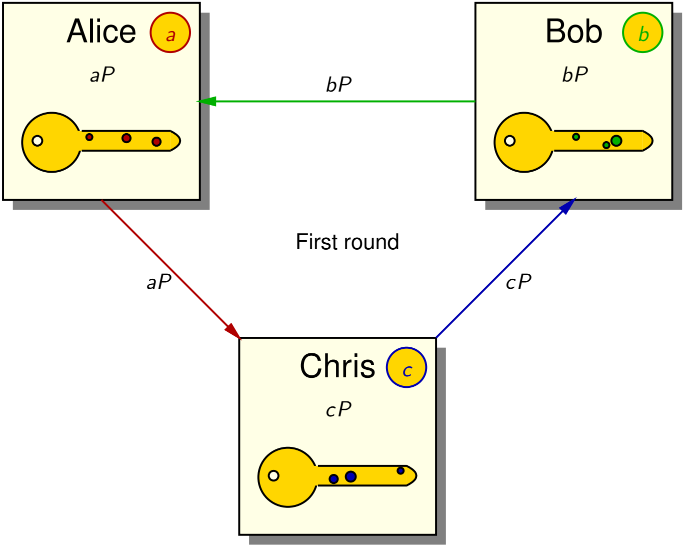




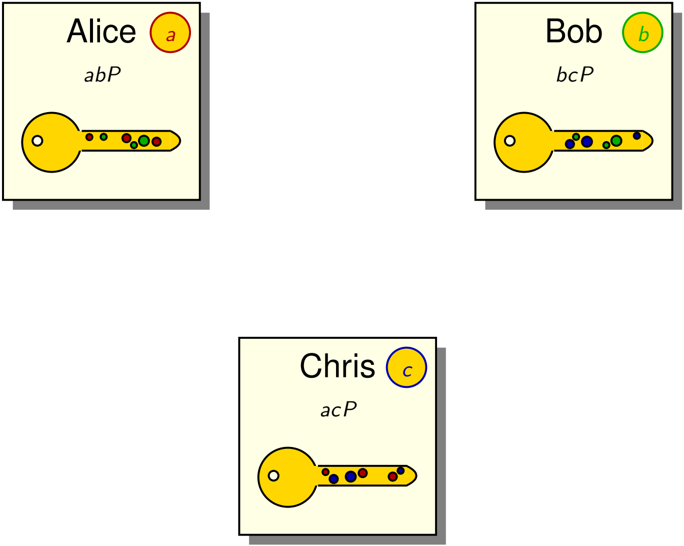
Example: Three-Party Key Agreement



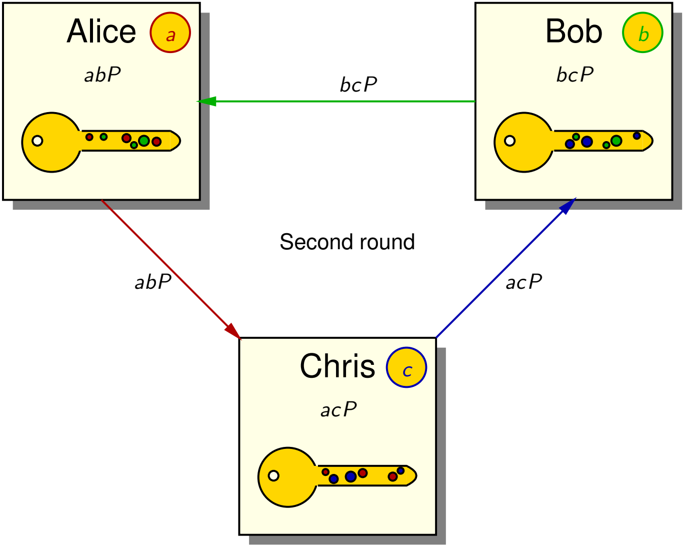
Example: Three-Party Key Agreement



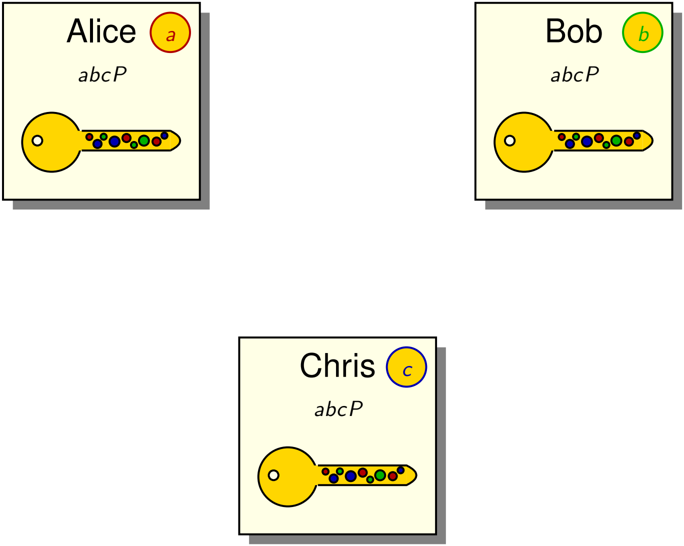
Example: Three-Party Key Agreement



Example: Three-Party Key Agreement



Example: Three-Party Key Agreement



Example: Three-Party Key Agreement

Three-party two-round key agreement protocol

Does a three-party one-round key agreement protocol exist?

Example: Three-Party Key Agreement

Bilinear pairing

- $G_1 = \langle P \rangle$: additively-written group
- G_2: multiplicatively-written group with identity 1
- A bilinear pairing on (G_1, G_2) is a map

$$\hat{e} : G_1 \times G_1 \longrightarrow G_2$$

that satisfies the following conditions:

1. **Bilinearity.** For all $Q, R, S \in G_1$,

$$\hat{e}(Q + R, S) = \hat{e}(Q, S)\hat{e}(R, S) \quad \text{and} \quad \hat{e}(Q, R + S) = \hat{e}(Q, R)\hat{e}(Q, S).$$

2. **Non-degeneracy.** $\hat{e}(P, P) \neq 1$.
3. **Computability.** $\hat{e}$ can be efficiently computed.

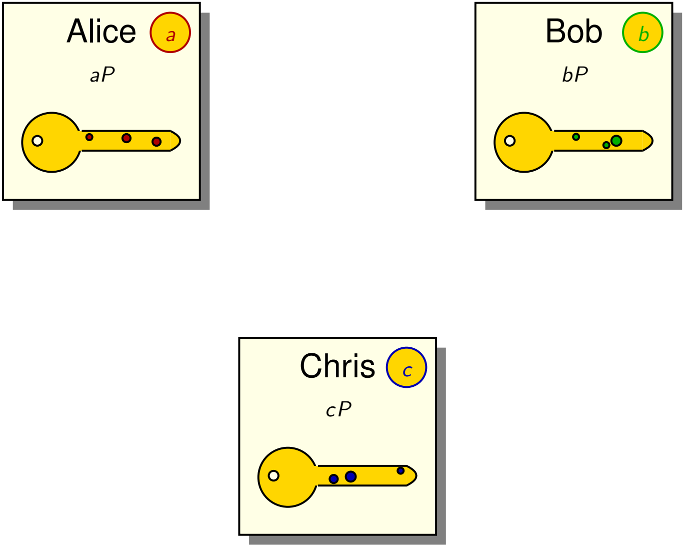
Example: Three-Party Key Agreement

Bilinear Diffie-Hellman problem (BDHP)

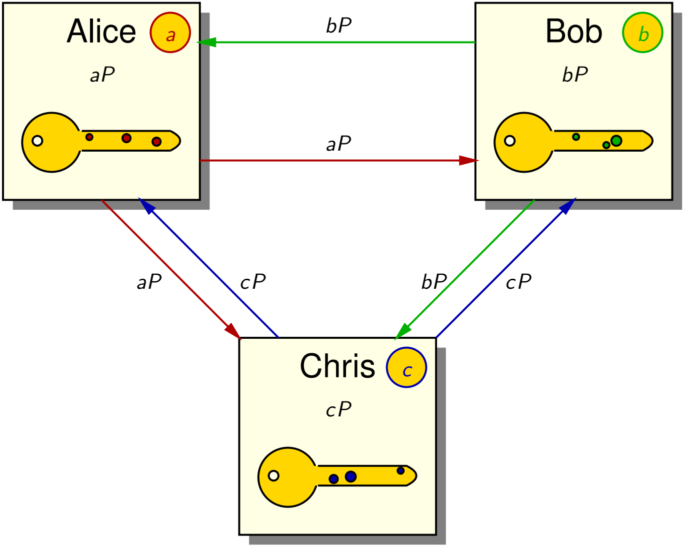
Given P, aP, bP, and cP, compute $\hat{e}(P, P)^{abc}$

Assumption: the BDHP is difficult

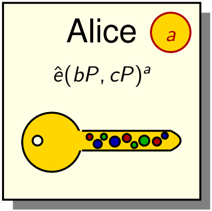
Example: Three-Party Key Agreement

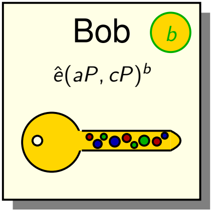


Example: Three-Party Key Agreement

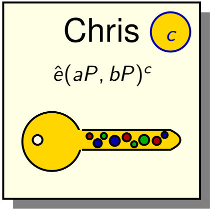


Example: Three-Party Key Agreement





$$\hat{e}(bP, cP)^a = \hat{e}(aP, cP)^b = \hat{e}(aP, bP)^c = \hat{e}(P, P)^{abc}$$



Example: Three-Party Key Agreement

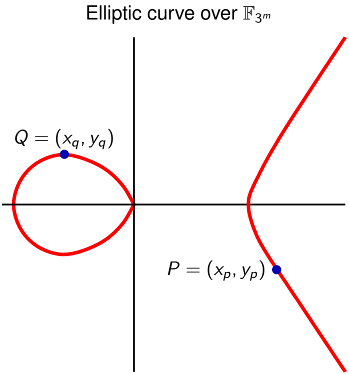
Examples of cryptographic bilinear maps

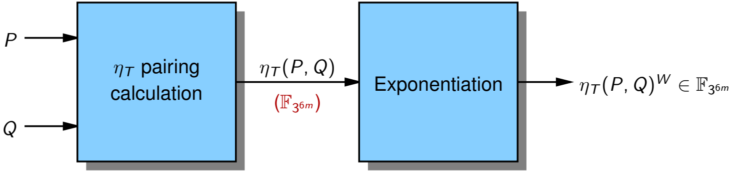
- Weil pairing
- Tate pairing
- η_T pairing (Barreto *et al.*)
- Ate pairing (Hess *et al.*)

Applications

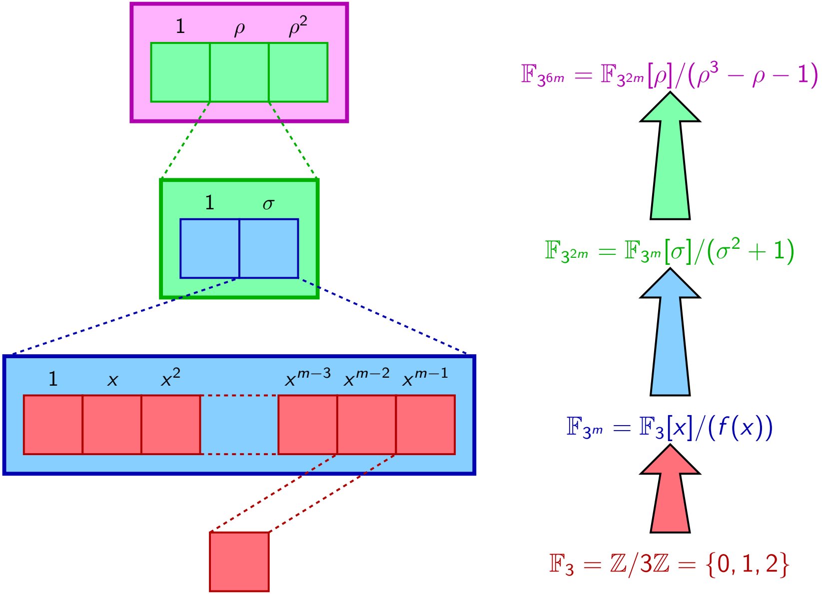
- Identity based encryption
- Short signature

Computation of the η_T Pairing

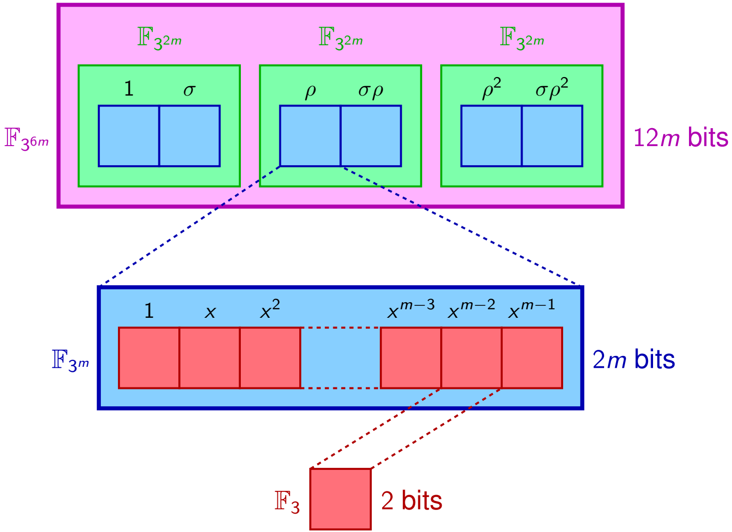




Computation of the η_T Pairing – Tower Field



Computation of the η_T Pairing – Tower Field



Computation of the η_T Pairing

$\eta_T(P, Q)$

- Addition
- Multiplication
- Cubing
- Cube root

$\eta_T(P, Q)^{3^{\frac{m+1}{2}}}$ (Arith 18)

- Addition
- Multiplication
- Cubing

Bilinearity of $\eta_T(P, Q)^W$

$$\eta_T\left(P, Q\right)^W = \sqrt[3^m]{\left(\eta_T\left(\left[3^{\frac{m-1}{2}}\right] P, Q\right)^{3^{\frac{m+1}{2}}}\right)^W}$$

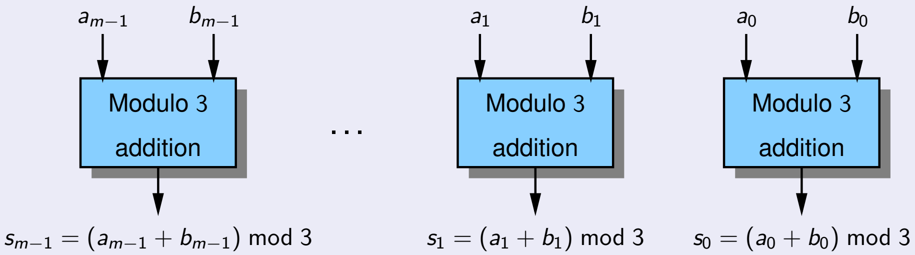
A Coprocessor for the Full Pairing Computation

Operations over $\mathbb{F}_{3^m}$

Additions	$51 \cdot \frac{m-1}{2} + 503$
Multiplications	$15 \cdot \frac{m-1}{2} + 86$
Cubings	$10m + 2$
Inversion	1

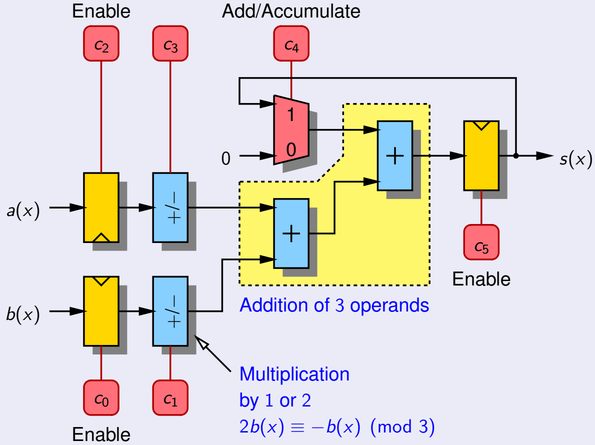
A Coprocessor for the Full Pairing Computation

Addition over $\mathbb{F}_{3^m}$



A Coprocessor for the Full Pairing Computation

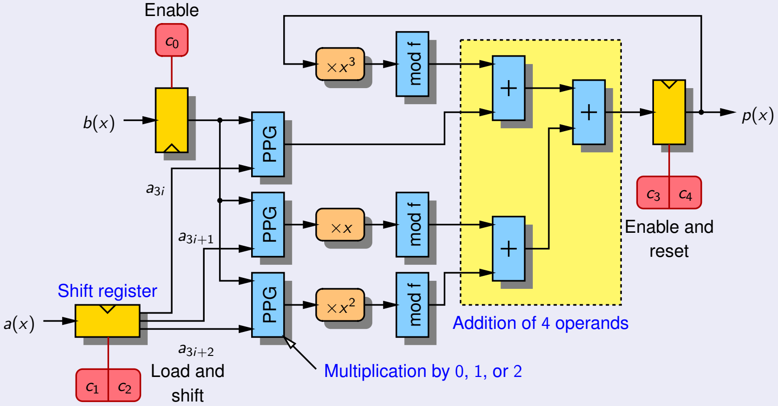
Addition, subtraction, and accumulation over $\mathbb{F}_{3^m}$



A Coprocessor for the Full Pairing Computation

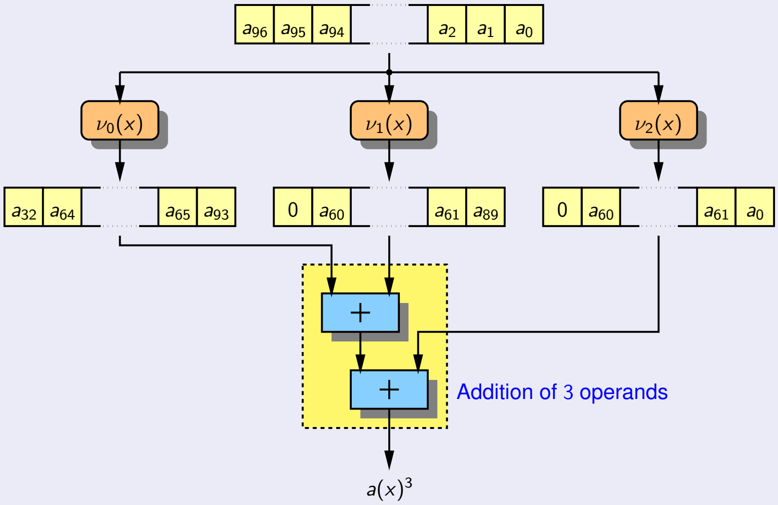
Multiplication over $\mathbb{F}_{3^m}$

- Array multiplier ($\lceil m/3 \rceil$ clock cycles)
- Most significant coefficient first (Horner's rule)



A Coprocessor for the Full Pairing Computation

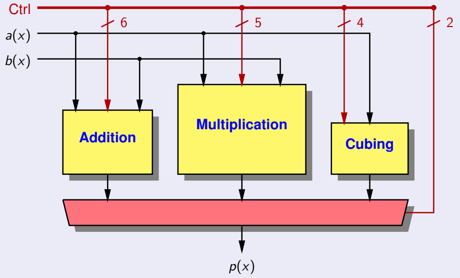
Cubing over $\mathbb{F}_3[x]/(x^{97} + x^{12} + 2)$



A Coprocessor for the Full Pairing Computation

Arithmetic operators over $\mathbb{F}_{3^{97}}$ on a Cyclone II FPGA

Operation	Area [LEs]	Control [bits]
Add./sub.	970	6
Mult.	1375	5
Cubing	668	4
ALU	3308	17



A Coprocessor for the Full Pairing Computation

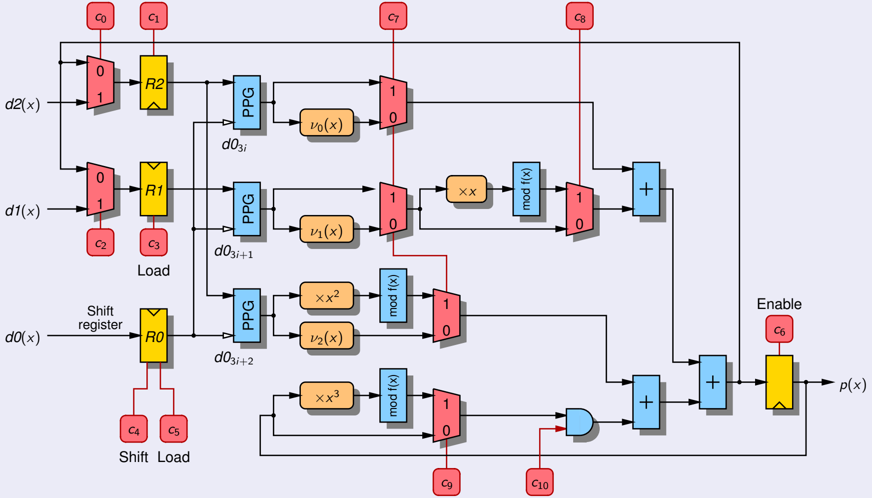
Unified arithmetic operator

- Operations
  - Addition
  - Subtraction
  - Accumulation
  - Multiplication
  - Cubing
- Area (Cyclone II): 2676 LEs (instead of 3308)
- Control bits: 11 (instead of 17)
- Inversion: Fermat's little theorem (96 cubings and 9 multiplications)

$$a^{3^m-2} = a^{-1}, \text{ where } a \in \mathbb{F}_{3^m}$$

A Coprocessor for the Full Pairing Computation

Unified arithmetic operator



A Coprocessor for the Full Pairing Computation

Results (CHES 2007)

- FPGA: Xilinx Virtex-II Pro 4
- $\mathbb{F}_3[x]/(x^{97} + x^{12} + 2)$
- Area: 1888 slices + 6 memory blocks
- Clock frequency: 147 MHz
- Clock cycles for a full pairing: 32618
- Calculation time: 222μs

A Coprocessor for the Full Pairing Computation

Results (CHES 2007)

- FPGA: Xilinx Virtex-II Pro 4
- $\mathbb{F}_3[x]/(x^{97} + x^{12} + 2)$
- Area: 1888 slices + 6 memory blocks
- Clock frequency: 147 MHz
- Clock cycles for a full pairing: 32618
- Calculation time: 222μs

Extended Euclidean algorithm (EEA)

- Area: 2210 additional slices
- Clock cycles for a full pairing: 32419 instead of 32618

Conclusion

Comparisons

Architecture	Area	Calculation time	FPGA
Arith 18 & Waifi 2007	18000 LEs	33 μs	Cyclone II
CHES 2007	1888 slices	222 μs	Virtex-II Pro
Grabher and Page (CHES 2005)	4481 slices	432 μs	Virtex-II Pro
Kerins *et al.* (CHES 2005)	55616 slices	850 μs	Virtex-II Pro
Ronan *et al.* (ITNG 2007)	10000 slices	178 μs	Virtex-II Pro

(1 slice $\approx$ 2 LEs)

Conclusion

VHDL code generator

- Generation of an unified operator according to $\mathbb{F}_{p^m}$ and $f(x)$
- Support for the following operations:
  - Addition
  - Multiplication
  - Frobenius ($a(x)^p \bmod f(x)$)
  - Inverse Frobenius ($\sqrt[p]{a(x)} \bmod f(x)$)

Conclusion

VHDL code generator

- Generation of an unified operator according to $\mathbb{F}_{p^m}$ and $f(x)$
- Support for the following operations:
  - Addition
  - Multiplication
  - Frobenius ($a(x)^p \bmod f(x)$)
  - Inverse Frobenius ($\sqrt[p]{a(x)} \bmod f(x)$)

Future work

- Automatic generation of the control unit
- Application (e.g. short signature)
- Genus 2
- Side-channel

A Coprocessor for the Full Pairing Computation