FPGA Implementations of Pairing using Residue Number System and Lazy Reduction

Ray Cheung [1] Sylvain Duquesne [2] Junfeng Fan [4]
Nicolas Guillermin [2,3] Ingrid Verbauwhede [4] Gavin Yao[1]

[1] Department of Electronic Engineering
City University of Hong Kong, Hong Kong SAR,

[2] IRMAR, UMR CNRS 6625, Université Rennes 1
Campus de Beaulieu, 35042 Rennes cedex, France

[3] DGA.IS, La Roche Marguerite - 35170 - Bruz, France

[4] ESAT/SCD-COSIC, Katholieke Universiteit Leuven
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium

Oct, 2011

Agenda

1. Motivations
2. Backgrounds
3. Pairing Coprocessor Design
4. Implementation Results
5. Conclusions

Motivations

Getting the fastest hardware implementation
for a 128 bit security Pairing

Motivations

Getting the fastest hardware implementation
for a 128 bit security Pairing

supersingular curves/small char.	ordinary curves/big char.

Motivations

Getting the fastest hardware implementation
for a 128 bit security Pairing

supersingular curves/small char.	ordinary curves/big char.
faster hardware architecture frobenius parallelism	

Motivations

Getting the fastest hardware implementation
for a 128 bit security Pairing

supersingular curves/small char.	ordinary curves/big char.
faster hardware architecture	faster software implementations
frobenius	embedding degree
parallelism	smaller curves

Motivations

Getting the fastest hardware implementation
for a 128 bit security Pairing

supersingular curves/small char.	ordinary curves/big char.
faster hardware architecture frobenius parallelism	faster software implementations embedding degree smaller curves

Question : Can we bridge the gap to make ordinary curves win?

Agenda

1. Motivations
2. Backgrounds
3. Pairing Coprocessor Design
4. Implementation Results
5. Conclusions

What is Pairing and how do we use it?

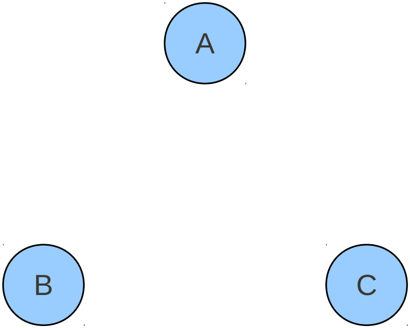
A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with $\mathbb{G}_1 \times \mathbb{G}_2$ and $\mathbb{G}_T$ groups with hard Discrete Logarithm

What is Pairing and how do we use it?

A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with $\mathbb{G}_1 \times \mathbb{G}_2$ and $\mathbb{G}_T$ groups with hard Discrete Logarithm

Useful for :

- Three-party one-round Diffie-Hellman key agreement [Joux'00]

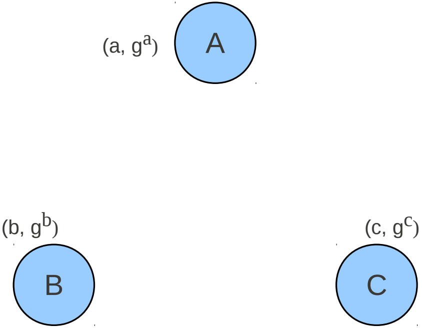


What is Pairing and how do we use it?

A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with $\mathbb{G}_1 \times \mathbb{G}_2$ and $\mathbb{G}_T$ groups with hard Discrete Logarithm

Useful for :

- Three-party one-round Diffie-Hellman key agreement [Joux'00]

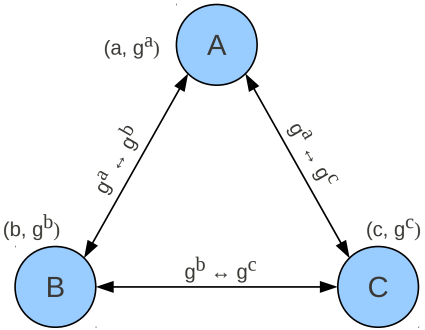


What is Pairing and how do we use it?

A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with $\mathbb{G}_1 \times \mathbb{G}_2$ and $\mathbb{G}_T$ groups with hard Discrete Logarithm

Useful for :

- Three-party one-round Diffie-Hellman key agreement [Joux'00]

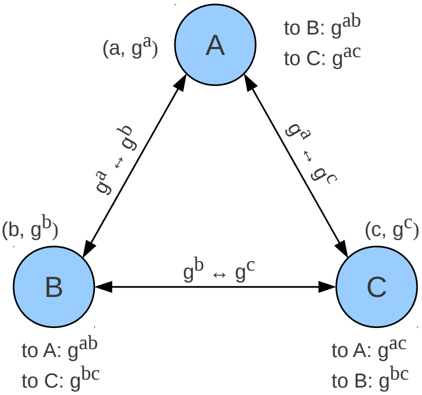


What is Pairing and how do we use it?

A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with $\mathbb{G}_1 \times \mathbb{G}_2$ and $\mathbb{G}_T$ groups with hard Discrete Logarithm

Useful for :

- Three-party one-round Diffie-Hellman key agreement [Joux'00]

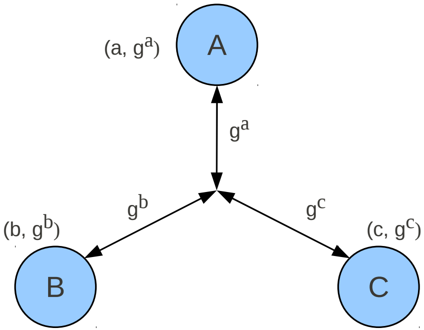


What is Pairing and how do we use it?

A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with $\mathbb{G}_1 \times \mathbb{G}_2$ and $\mathbb{G}_T$ groups with hard Discrete Logarithm

Useful for :

- Three-party one-round Diffie-Hellman key agreement [Joux'00]

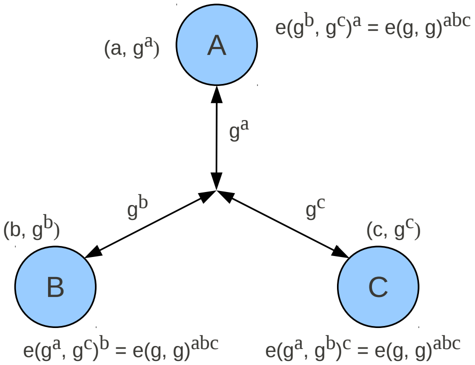


What is Pairing and how do we use it?

A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with $\mathbb{G}_1 \times \mathbb{G}_2$ and $\mathbb{G}_T$ groups with hard Discrete Logarithm

Useful for :

- Three-party one-round Diffie-Hellman key agreement [Joux'00]



What is Pairing and how do we use it?

A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with $\mathbb{G}_1 \times \mathbb{G}_2$ and $\mathbb{G}_T$ groups with hard Discrete Logarithm

Useful for :

- Three-party one-round Diffie-Hellman key agreement [Joux'00]
- Identity-based encryption [Boneh+01]
- Short signature [Boneh+01]
- Blind signature [Boldyreva'03]

Barreto-Naehrig Curves

BN curve over $\mathbb{F}_p$:

$$y^2 = x^3 + b$$

where $b \neq 0$ such that $\#E = \ell$

$$p = 36u^4 + 36u^3 + 24u^2 + 6u + 1$$

$$\ell = 36u^4 + 36u^3 + 18u^2 + 6u + 1$$

for $u \in \mathbb{Z}$ and p, ℓ primes.

Barreto-Naehrig Curves

BN curve over $\mathbb{F}_p$:

$$y^2 = x^3 + b$$

where $b \neq 0$ such that $\#E = \ell$

$$p = 36u^4 + 36u^3 + 24u^2 + 6u + 1$$

$$\ell = 36u^4 + 36u^3 + 18u^2 + 6u + 1$$

for $u \in \mathbb{Z}$ and p, ℓ primes.

Very adapted for 128 bits security

Pairing Parameter Selection

Chosen curves

Security	u	$\lceil \log_2 p \rceil$
126-bit	$-(2^{62} + 2^{55} + 1)$	254
128-bit	$-(2^{63} + 2^{22} + 2^{18} + 2^{7} + 1)$	258
192-bit	$-(2^{160} + 2^{74} + 2^{12} + 1)$	646

Optimal Ate Pairing on BN Curves

Require:
$P \in E(\mathbb{F}_p)[\ell]$, $Q = (x_Q\gamma^2, y_Q\gamma^3) \in E(\mathbb{F}_{p^{12}}) \cap \text{Ker}(\pi_p - p)$
with $x_Q, y_Q \in \mathbb{F}_{p^2}$, $r = |6u+2| = \sum_{i=0}^{s-1} r_i 2^i$, where $u < 0$.

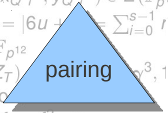
Ensure: $a_{opt}(Q, P) \in \mathbb{F}_{p^{12}}$

$T = (X_T\gamma^2, Y_T\gamma^3, Z_T) \leftarrow (x_Q\gamma^2, y_Q\gamma^3, 1), f \leftarrow 1$
for $i = s-2$ downto 0 **do**
  $T, g \leftarrow \text{dbl}(T, P), f \leftarrow f^2 \cdot g$
  if $r_i = 1$ **then**
    $T, g \leftarrow \text{add}(T, Q, P), f \leftarrow f \cdot g$
  end if
end for
$T \leftarrow -T, f \leftarrow f^{p^6}$ (f^{p^6} is equivalent to f^{-1})
$Q_1 \leftarrow \pi_p(Q), Q_2 \leftarrow -\pi_p(Q_1)$
$T, g \leftarrow \text{add}(T, Q_1, P), f \leftarrow f \cdot g$
$T, g \leftarrow \text{add}(T, Q_2, P), f \leftarrow f \cdot g$
$f \leftarrow \left(f^{p^6-1}\right)^{p^2+1}$
$f \leftarrow f^{(p^4-p^2+1)/\ell}$
return f

Optimal Ate Pairing on BN Curves

Require:
$P \in E(\mathbb{F}_p)[\ell]$, $Q = (x_Q\gamma^2, y_Q\gamma^3) \in E(\mathbb{F}_{p^{12}}) \cap \mathrm{Ker}(\pi_p - p)$
with $x_Q, y_Q \in \mathbb{F}_{p^2}$, $r = |6u + \ldots = \sum_{i=0}^{s-1} r \ldots$

Ensure: $a_{opt}(Q, P) \in \mathbb{F}_{p^{12}}$



Tate [Frey[+]94]
Ate [Hess[+]06]
Optimal [Lee[+]08,Vercauteren'10]

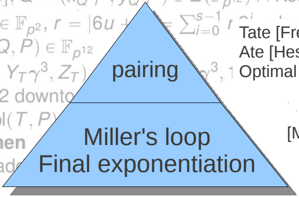
$T = (X_T\gamma^2, Y_T\gamma^3, Z_T) \ldots \gamma^3, 1 \ldots$
for $i = s - 2$ downto ...
  $T, g \leftarrow \mathrm{dbl}(T, P)$, $f \leftarrow f^2 \cdot g$
  if $r_i = 1$ **then**
    $T, g \leftarrow \mathrm{add}(T, Q, P)$, $f \leftarrow f \cdot g$
  end if
end for
$T \leftarrow -T$, $f \leftarrow f^{p^6}$ (f^{p^6} is equivalent to f^{-1})
$Q_1 \leftarrow \pi_p(Q)$, $Q_2 \leftarrow -\pi_p(Q_1)$
$T, g \leftarrow \mathrm{add}(T, Q_1, P)$, $f \leftarrow f \cdot g$
$T, g \leftarrow \mathrm{add}(T, Q_2, P)$, $f \leftarrow f \cdot g$
$f \leftarrow \left(f^{p^6-1}\right)^{p^2+1}$
$f \leftarrow f^{(p^4-p^2+1)/\ell}$
return f

Optimal Ate Pairing on BN Curves

Require:
$P \in E(\mathbb{F}_p)[\ell]$, $Q = (x_Q\gamma^2, y_Q\gamma^3) \in E(\mathbb{F}_{p^{12}}) \cap \mathrm{Ker}(\pi_p - p)$
with $x_Q, y_Q \in \mathbb{F}_{p^2}$, $r = |6u +$ … $= \sum_{i=0}^{s-1} r$ …

Ensure: $a_{opt}(Q,P) \in \mathbb{F}_{p^{12}}$

$T = (X_T\gamma^2, Y_T\gamma^3, Z_T)$ … γ^3, …
for $i = s-2$ downto …
  $T, g \leftarrow \mathrm{dbl}(T, P)$ …
  if $r_i = 1$ **then**
    $T, g \leftarrow$ ad…
  end if
end for
$T \leftarrow -T, f \leftarrow f^{p^6}$ (f^{p^6} is equivalent to f^{-1})
$Q_1 \leftarrow \pi_p(Q), Q_2 \leftarrow -\pi_p(Q_1)$
$T, g \leftarrow \mathrm{add}(T, Q_1, P), f \leftarrow f \cdot g$
$T, g \leftarrow \mathrm{add}(T, Q_2, P), f \leftarrow f \cdot g$
$f \leftarrow \left(f^{p^6-1}\right)^{p^2+1}$
$f \leftarrow f^{(p^4-p^2+1)/\ell}$
return f



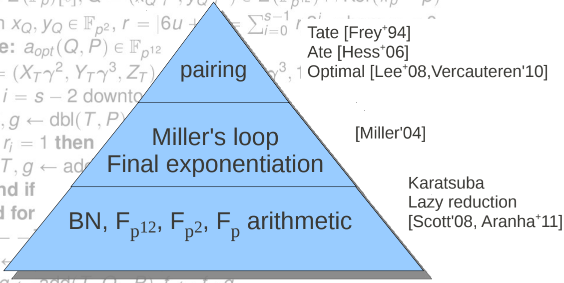
Tate [Frey+94]
Ate [Hess+06]
Optimal [Lee+08,Vercauteren'10]

[Miller'04]

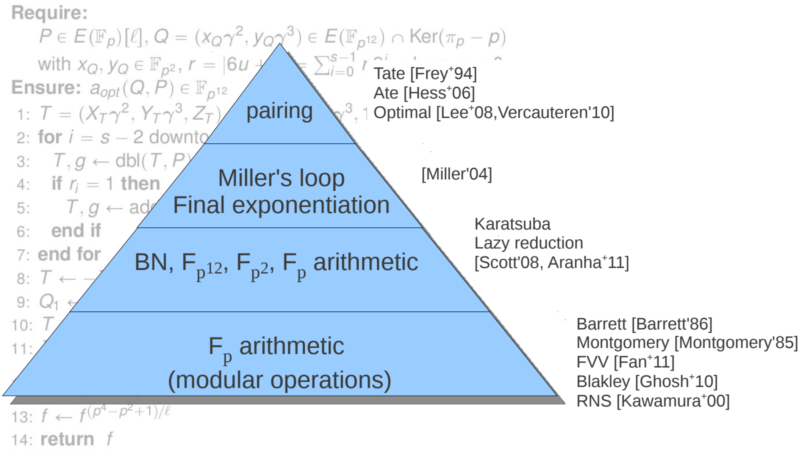
Optimal Ate Pairing on BN Curves

Require:
$P \in E(\mathbb{F}_p)[\ell]$, $Q = (x_Q\gamma^2, y_Q\gamma^3) \in E(\mathbb{F}_{p^{12}}) \cap \mathrm{Ker}(\pi_p - p)$
with $x_Q, y_Q \in \mathbb{F}_{p^2}$, $r = |6u + \ldots = \sum_{i=0}^{s-1} r \ldots$
Ensure: $a_{opt}(Q, P) \in \mathbb{F}_{p^{12}}$

$T = (X_T\gamma^2, Y_T\gamma^3, Z_T) \ldots$
for $i = s-2$ downto ...
  $T, g \leftarrow \mathrm{dbl}(T, P) \ldots$
  if $r_i = 1$ **then**
    $T, g \leftarrow$ add ...
  end if
end for
$T \leftarrow \ldots$
$Q_1 \leftarrow \ldots$
$T, g \leftarrow \mathrm{add}(T, Q_1, P), f \leftarrow f \cdot g$
$T, g \leftarrow \mathrm{add}(T, Q_2, P), f \leftarrow f \cdot g$
$f \leftarrow \left(f^{p^6-1}\right)^{p^2+1}$
$f \leftarrow f^{(p^4-p^2+1)/\ell}$
return f



Optimal Ate Pairing on BN Curves



Residue Number System (RNS)

RNS is defined by n pairwise coprime integer constants:

$$\mathfrak{B} = \{b_1, b_2, \cdots, b_n\}.$$

$$M_{\mathfrak{B}} := \prod_{i=1}^{n} b_i, b_i \in \mathfrak{B}$$

Residue Number System (RNS)

RNS is defined by n pairwise coprime integer constants:

$$\mathfrak{B} = \{b_1, b_2, \cdots, b_n\}.$$

$$M_{\mathfrak{B}} := \prod_{i=1}^{n} b_i, b_i \in \mathfrak{B}$$

Any integer $X, 0 \leqslant X < M_{\mathfrak{B}}$, X is uniquely represented by:

$$\mathfrak{X} = \{X \mod b_1, X \mod b_2, \cdots, X \mod b_n\},$$

Residue Number System (RNS)

RNS is defined by n pairwise coprime integer constants:

$$\mathfrak{B} = \{b_1, b_2, \cdots, b_n\}.$$

$$M_{\mathfrak{B}} := \prod_{i=1}^{n} b_i, b_i \in \mathfrak{B}$$

Any integer $X, 0 \leqslant X < M_{\mathfrak{B}}$, X is uniquely represented by:

$$\mathfrak{X} = \{X \mod b_1, X \mod b_2, \cdots, X \mod b_n\},$$

Arithmetic operations on RNS ($\mathbb{Z}/M_{\mathfrak{B}}\mathbb{Z}$)

Normal	RNS
$R = X \pm Y \bmod M_{\mathfrak{B}}$	$r_i = x_i \pm y_i \mod b_i$
$R = X \cdot Y \bmod M_{\mathfrak{B}}$	$r_i = x_i \cdot y_i \mod b_i$
$R = X/Y \bmod M_{\mathfrak{B}}$	$r_i = x_i y_i^{-1} \mod b_i$

only if $\gcd(Y, M_{\mathfrak{B}}) = 1$

RNS Montgomery reduction [Bajard+98]

RNS Montgomery – $M_{\mathfrak{B}}$

Input: $A = aM_{\mathfrak{B}} \bmod p$ and
$B = bM_{\mathfrak{B}} \bmod p$

Output: $T = abM_{\mathfrak{B}} \bmod p$

in $\mathfrak{B}$

1: $T_{\mathfrak{B}} \leftarrow A_{\mathfrak{B}} B_{\mathfrak{B}}$

2: $Q_{\mathfrak{B}} \leftarrow T_{\mathfrak{B}} \cdot (-p)^{-1}$

3: $S_{\mathfrak{B}} \leftarrow (T + Q_{\mathfrak{B}} p)/M_{\mathfrak{B}}$

RNS Montgomery reduction [Bajard+98]

RNS Montgomery – $M_{\mathfrak{B}}$

Input: $A = aM_{\mathfrak{B}} \bmod p$ and
$B = bM_{\mathfrak{B}} \bmod p$

Output: $T = abM_{\mathfrak{B}} \bmod p$

in $\mathfrak{B}$

1: $T_{\mathfrak{B}} \leftarrow A_{\mathfrak{B}} B_{\mathfrak{B}}$

2: $Q_{\mathfrak{B}} \leftarrow T_{\mathfrak{B}} \cdot (-p)^{-1}$

3: $S_{\mathfrak{B}} \leftarrow (T + Q_{\mathfrak{B}} p)/M_{\mathfrak{B}}$

$\gcd(M_{\mathfrak{B}}, M_{\mathfrak{B}}) = M_{\mathfrak{B}} \neq 1$

$M_{\mathfrak{B}}^{-1}$ does not exist in $\mathfrak{B}$.

RNS Montgomery reduction [Bajard+98]

RNS Montgomery – $M_{\mathfrak{B}}$

Input: $A = aM_{\mathfrak{B}} \bmod p$ and
$B = bM_{\mathfrak{B}} \bmod p$

Output: $T = abM_{\mathfrak{B}} \bmod p$

	in $\mathfrak{B}$	in $\mathfrak{C}$
1:	$T_{\mathfrak{B}} \leftarrow A_{\mathfrak{B}} B_{\mathfrak{B}}$	$T_{\mathfrak{C}} \leftarrow A_{\mathfrak{C}} B_{\mathfrak{C}}$
2:	$Q_{\mathfrak{B}} \leftarrow T_{\mathfrak{B}} \cdot (-p)^{-1}$	
4:		$S_{\mathfrak{C}} \leftarrow (T_{\mathfrak{C}} + Q_{\mathfrak{C}} p)(M_{\mathfrak{B}}^{-1})_{\mathfrak{C}}$

Introduce a new base $\mathfrak{C}$ to perform division by $M_{\mathfrak{B}}$.

RNS Montgomery reduction [Bajard+98]

RNS Montgomery – $M_{\mathfrak{B}}$

Input: $A = aM_{\mathfrak{B}} \bmod p$ and
$B = bM_{\mathfrak{B}} \bmod p$

Output: $T = abM_{\mathfrak{B}} \bmod p$

	in $\mathfrak{B}$		in $\mathfrak{C}$
1:	$T_{\mathfrak{B}} \leftarrow A_{\mathfrak{B}} B_{\mathfrak{B}}$		$T_{\mathfrak{C}} \leftarrow A_{\mathfrak{C}} B_{\mathfrak{C}}$
2:	$Q_{\mathfrak{B}} \leftarrow T_{\mathfrak{B}} \cdot (-p)^{-1}$		
3:	$Q_{\mathfrak{B}}$	$\xrightarrow{\text{Base Extension}}$	$Q_{\mathfrak{C}}$
4:			$S_{\mathfrak{C}} \leftarrow (T_{\mathfrak{C}} + Q_{\mathfrak{C}} p)(M_{\mathfrak{B}}^{-1})_{\mathfrak{C}}$
5:	$S_{\mathfrak{B}}$	$\xleftarrow{\text{Base Extension}}$	$S_{\mathfrak{C}}$

Introduce a new base $\mathfrak{C}$ to perform division by $M_{\mathfrak{B}}$.

Needs the computation of Base Extension [Kawamura+00]

RNS complexity and Lazy reduction

RNS Montgomery

Multiplication :

$2n$ MUL

Reduction (RED):

$2n^2 + 3n$ MUL

$AB \mod p$:

$2n^2 + 5n$ MUL

Conventional Montgomery

Multiplication:

n^2 MUL

Reduction:

$n^2 + n$ MUL

$AB \mod p$:

$2n^2 + n$ MUL

RNS complexity and Lazy reduction

RNS Montgomery

Multiplication :

$2n$ MUL

Reduction (RED):

$2n^2 + 3n$ MUL

Conventional Montgomery

Multiplication:

n^2 MUL

Reduction:

$n^2 + n$ MUL

RNS Montgomery:

$AB \mod p$:

$2n^2 + 5n$ MUL

$AB + CD \mod p$:

$2n^2 + 7n$ MUL

Conventional Montgomery:

$AB \mod p$:

$2n^2 + n$ MUL

$AB + CD \mod p$:

$3n^2 + n$ MUL

RNS complexity and Lazy reduction

RNS Montgomery	Conventional Montgomery
Multiplication : $2n$ MUL	Multiplication: n^2 MUL
Reduction (RED): $2n^2 + 3n$ MUL	Reduction: $n^2 + n$ MUL
$AB \mod p$: $2n^2 + 5n$ MUL	$AB \mod p$: $2n^2 + n$ MUL
$AB + CD \mod p$: $2n^2 + 7n$ MUL	$AB + CD \mod p$: $3n^2 + n$ MUL
$\sum_{i=1}^{k} A_i B_i \mod p$: $2n^2 + (3 + 2k)n$ MUL	$\sum_{i=1}^{k} A_i B_i \mod p$: $(1 + k)n^2 + n$ MUL

Theoretical conclusions

- Lazy reduction reduces the complexity of Pairings [Aranha+11]

Theoretical conclusions

- Lazy reduction reduces the complexity of Pairings [Aranha+11]
- RNS reduces the complexity of lazy reduction

Theoretical conclusions

- Lazy reduction reduces the complexity of Pairings [Aranha+11]
- RNS reduces the complexity of lazy reduction
- RNS involves easy parallelism

Theoretical conclusions

- Lazy reduction reduces the complexity of Pairings [Aranha+11]
- RNS reduces the complexity of lazy reduction
- RNS involves easy parallelism

Remains to verify it "in real world"

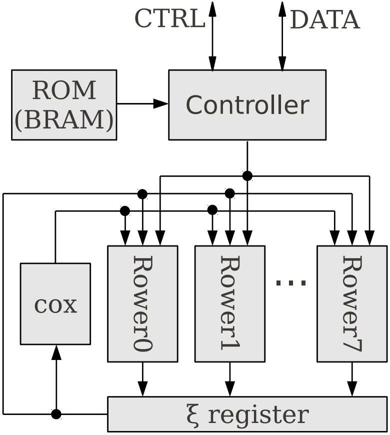
Agenda

1. Motivations
2. Backgrounds
3. Pairing Coprocessor Design
4. Implementation Results
5. Conclusions

Cox-Rower Architecture [Kawamura+00]

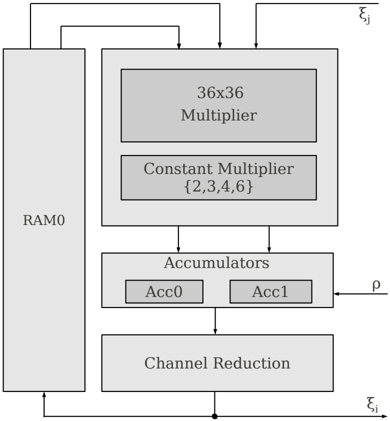
Main feature

- n rowers
  - MUL: 2 cycles
  - RED: 2n+3 cycles
- One rower, one channel
- Microcoded sequencer



Rower Design

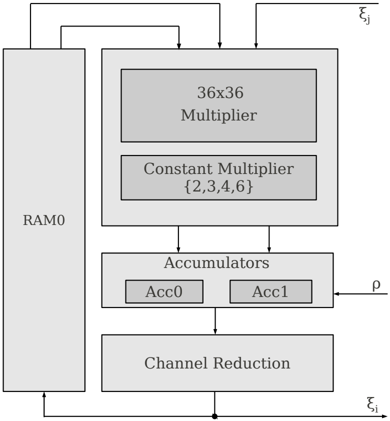
- 2 accumulators
- Small-constant multiplier
- 3-port RAMs
- Same data path



Rower Design

Underlying Field

- $\mathbb{F}_{p^2} = \mathbb{F}_p[\mathbf{i}]/(\mathbf{i}^2 + 1)$
- $\mathbb{F}_{p^{12}} = \mathbb{F}_{p^2}[\gamma]/(\gamma^6 - (1 + \mathbf{i}))$

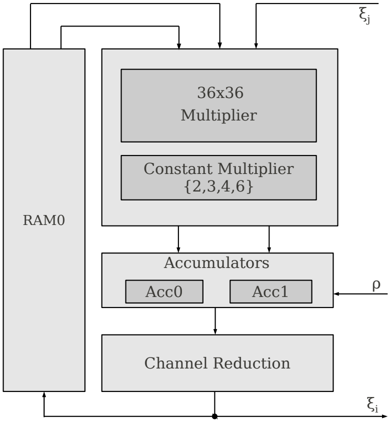


Rower Design

Underlying Field

- $\mathbb{F}_{p^2} = \mathbb{F}_p[\mathbf{i}]/(\mathbf{i}^2 + 1)$
- $\mathbb{F}_{p^{12}} = \mathbb{F}_{p^2}[\gamma]/(\gamma^6 - (1 + \mathbf{i}))$

$$(x_0 + x_1\mathbf{i})(y_0 + y_1\mathbf{i})(1 + \mathbf{i})$$

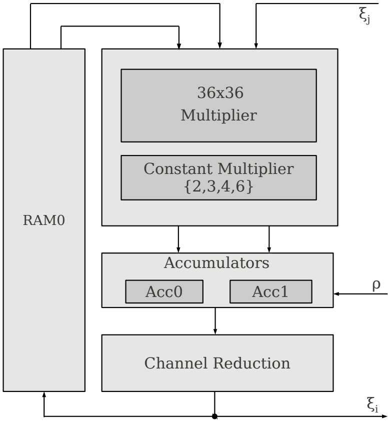


Rower Design

Underlying Field

- $\mathbb{F}_{p^2} = \mathbb{F}_p[\mathbf{i}]/(\mathbf{i}^2 + 1)$
- $\mathbb{F}_{p^{12}} = \mathbb{F}_{p^2}[\gamma]/(\gamma^6 - (1 + \mathbf{i}))$

$$
\begin{aligned}
&(x_0 + x_1\mathbf{i})(y_0 + y_1\mathbf{i})(1 + \mathbf{i}) \\
&= (x_0y_0 - x_1y_1 - x_0y_1 - x_1y_0) + \\
&\quad (x_0y_0 - x_1y_1 + x_0y_1 + x_1y_0)\mathbf{i}
\end{aligned}
$$



Further Optimization

Observations

- Computational intensiveness: base extension.

Further Optimization

Observations

- Computational intensiveness: base extension.
- Base extension: n^2 multiplications by constant.

Further Optimization

Observations

- Computational intensiveness: base extension.
- Base extension: n^2 multiplications by constant.
- Constant is determined by base $\mathfrak{B}$ and $\mathfrak{C}$.

Further Optimization

Observations

- Computational intensiveness: base extension.
- Base extension: n^2 multiplications by constant.
- Constant is determined by base $\mathfrak{B}$ and $\mathfrak{C}$.

- Constant size is 35 bits.

Further Optimization

Observations

- Computational intensiveness: base extension.
- Base extension: n^2 multiplications by constant.
- Constant is determined by base $\mathfrak{B}$ and $\mathfrak{C}$.

- Constant size is 35 bits.
- With selected bases, bit-length: $35 \rightarrow 25$.

Further Optimization

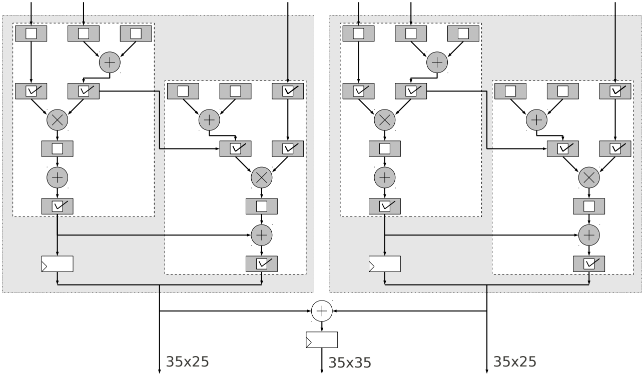
Observations

- Computational intensiveness: base extension.
- Base extension: n^2 multiplications by constant.
- Constant is determined by base $\mathfrak{B}$ and $\mathfrak{C}$.

- Constant size is 35 bits.
- With selected bases, bit-length: $35 \rightarrow 25$.
- Bit-length of the constant is shortened.

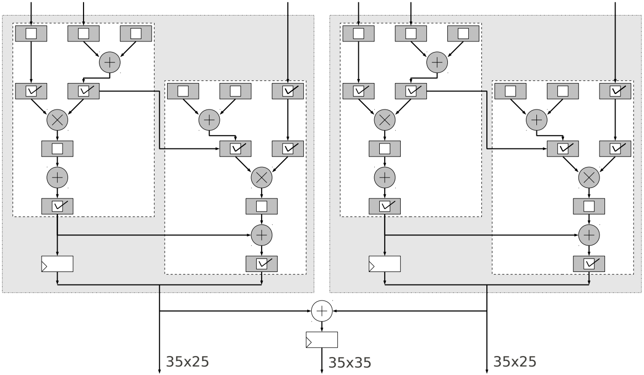
Dual mode 4-stage pipelined MUL

- One 35×35 multiplier
- Two 35×25 multipliers
- @250MHz on Virtex-6
- MUL: 2 cycles



Dual mode 4-stage pipelined MUL

- One 35×35 multiplier
- Two 35×25 multipliers
- @250MHz on Virtex-6
- MUL: 2 cycles
- RED: n+4 cycles (v.s. 2n+3 cycles)



Agenda

Pairing Parameter Selection

Design I

Security	u	$\lceil \log_2 p \rceil$
126-bit	$-(2^{62}+2^{55}+1)$	254
128-bit	$-(2^{63}+2^{22}+2^{18}+2^{7}+1)$	258
192-bit	$-(2^{160}+2^{74}+2^{12}+1)$	646

Design II

126-bit	$-(2^{62}+2^{55}+1)$	254

Logic Utilization and Cycle Count

Logic utilization

Design	*n*	**Device**	**Multipliers**	**Logic Elements**	**Embedded Memory**
I (Altera)	8	Cyclone II	35 18multipliers	14274 LC	67 M4k
	8	Stratix III	72 DSP18el	4233 ALMs	1 M144k + 18 M9k
	19	Stratix III	171 DSP18el	9910 ALMs	1 M144k + 40 M9k
II (Xilinx)	8	Virtex-6	32 DSP48E1s	7032 Slices	45 18Kb BRAMs

Logic Utilization and Cycle Count

Logic utilization

Design	n	Device	Multipliers	Logic Elements	Embedded Memory
I (Altera)	8	Cyclone II	35 18multipliers	14274 LC	67 M4k
	8	Stratix III	72 DSP18el	4233 ALMs	1 M144k + 18 M9k
	19	Stratix III	171 DSP18el	9910 ALMs	1 M144k + 40 M9k
II (Xilinx)	8	Virtex-6	32 DSP48E1s	7032 Slices	45 18Kb BRAMs

Cycle count and Latency

	Curve	Cycles	Technology	Frequency	Latency
Design I	BN_{126}	176111	Cyclone II	91 MHz	1.93 ms
	BN_{126}	176111	Stratix III	165 MHz	1.07 ms
	BN_{128}	192502	Stratix III	165 MHz	1.16 ms
	BN_{192}	789849	Stratix III	131 MHz	6.02 ms
Design II	BN_{126}	143111	Virtex-6	250 MHz	0.57 ms

Comparison

Design	Pairing/ Security[bit]	Platform	Algorithm	Area	Freq. [MHz]	Cycle	Delay [ms]
Design I	optimal ate 126	Altera (Stratix III)	RNS (Parallel)	4233 ALMs 72 DSPs	165	176,111	**1.07**
Design II	optimal ate 126	Xilinx (Virtex-6)	RNS (Parallel)	7032 slices 32 DSPs	250	143,111	**0.573**
Fan^{+}11	ate/128	Xilinx (Virtex-6)	HMM (Parallel)	4014 slices 42 DSPs	210	336,366	1.60
	opt. ate/128					245,430	1.17
Estibals'10	Tate $\mathbb{F}_{3^{5\cdot 97}}$ 128	Xilinx (Virtex-4)	-	4755 Slices 7 BRAMs	192	428,853	2.23
Aranha^{+}10	opt. Eta $\mathbb{F}_{2^{367}}$ 128	Xilinx (Virtex-4)	-	4518 Slices	220	773,960*	3.52
Ghosh^{+}11	$\eta_T \mathbb{F}_{2^{1223}}$ 128	Xilinx (Virtex-6)	-	15167 Slices	250	76,000*	0.19
Beuchat^{+}10	optimal ate 126	Core i7	Montgomery	-	2800	2,330,000	0.83
Aranha^{+}11	optimal ate 126	Phenom II	Montgomery	-	3000	1,562,000	0.52

Conclusions

Conclusions

1. Pairing using RNS + lazy reduction.
2. Novel base selection specification.
3. Hardware architectures.
4. New speed record.

Thank you!