Co-Z Addition Formulæ and Binary Ladders on Elliptic Curves

Raveen Goundar ▪ Marc Joye ▪ Atsuko Miyaji

technicolor

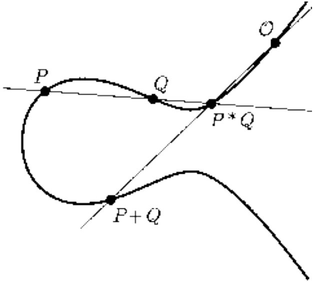
Co-Z Addition Formulæ and Binary Ladders on Elliptic Curves

Raveen Goundar ▪ Marc Joye ▪ Atsuko Miyaji

technicolor

Elliptic Curve Cryptography

- Invented [independently] by Neil Koblitz and Victor Miller in 1985



- Useful for key exchange, encryption and digital signature

Scalar Multiplication

Definition

Given scalar k and a point $\boldsymbol{P}$, compute $[k]\boldsymbol{P} = \underbrace{\boldsymbol{P} + \boldsymbol{P} + \cdots + \boldsymbol{P}}_{k \text{ times}}$

ECDLP Given $\boldsymbol{P}$ and $\boldsymbol{Q} = [k]\boldsymbol{P}$, recover k

- no subexponential algorithms are known to solve the ECDLP (in the *general* case)
- smaller key sizes can be used

	Bit security				
	80	112	128	192	256
ECC	160	224	256	384	512
RSA	1024	2048	3072	8192	15360

technicolor

This Talk

Goal

Implementation of the Montgomery ladder and of its dual version using efficient co-Z formulæ

- binary scalar multiplication algorithms
- suitable for memory-constrained devices

technicolor

Outline

1. **Arithmetic on Elliptic Curves**
   - Jacobian coordinates
   - Co-Z point addition
2. **Binary Scalar Multiplication Algorithms**
   - Left-to-right methods
   - Right-to-left methods
3. **New Implementations**
   - Binary ladders with co-Z trick
   - Point double-add operation
4. **Discussion**
   - Performance analysis
   - Security analysis
5. **Conclusion**

technicolor

Outline

1 **Arithmetic on Elliptic Curves**
- Jacobian coordinates
- Co-Z point addition

2 **Binary Scalar Multiplication Algorithms**
- Left-to-right methods
- Right-to-left methods

3 **New Implementations**
- Binary ladders with co-Z trick
- Point double-add operation

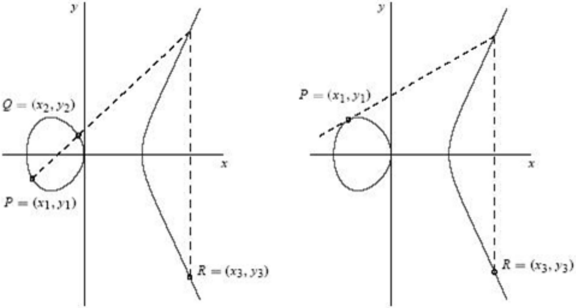
4 **Discussion**
- Performance analysis
- Security analysis

5 **Conclusion**

Elliptic Curves

Weierstraß equation (affine coordinates)

Let $E : y^2 = x^3 + ax + b$ define over $\mathbb{F}_q$ ($char \neq 2, 3$) with discriminant $\Delta = -16(4a^3 + 27b^2) \neq 0$



(a) Addition: $P + Q = R$. (b) Doubling: $P + P = R$.

technicolor

Jacobian Coordinates

- To avoid computation of inverse in $\mathbb{F}_q$
  - affine point $(x, y) \rightarrow$ projective point $(X : Y : Z)$ such that $x = X/Z^2$ and $y = Y/Z^3$

Weierstraß equation (projective Jacobian coordinates)

Let $E : Y^2 = X^3 + aXZ^4 + bZ^6$ define over $\mathbb{F}_q$ $(char \neq 2, 3)$ with discriminant $\Delta = -16(4a^3 + 27b^2) \neq 0$

- Point at infinity $\boldsymbol{O} = (1 : 1 : 0)$
- If $\boldsymbol{P} = (X_1 : Y_1 : Z_1) \in E$ then $-\boldsymbol{P} = (X_1 : -Y_1 : Z_1)$

technicolor

Co-Z Point Addition (ZADD)

- Introduced by Meloni [WAIFI 2007]
- Addition of two distinct points with the same Z-coordinate

Co-Z point addition

Let $\boldsymbol{P} = (X_1 : Y_1 : Z)$ and $\boldsymbol{Q} = (X_2 : Y_2 : Z)$. Then $\boldsymbol{P} + \boldsymbol{Q} = (X_3 : Y_3 : Z_3)$ where

$$X_3 = D - W_1 - W_2, \quad Y_3 = (Y_1 - Y_2)(W_1 - X_3) - A_1, \quad Z_3 = Z(X_1 - X_2)$$

with $A_1 = Y_1(W_1 - W_2)$, $W_1 = X_1 C$, $W_2 = X_2 C$, $C = (X_1 - X_2)^2$ and $D = (Y_1 - Y_2)^2$

- Cost of ZADD: $5\mathrm{M} + 2\mathrm{S}$

technicolor

Co-Z Point Addition with Update (ZADDU)

- Main advantage of Meloni's addition

Equivalent representation of P

Evaluation of $\boldsymbol{R} = \mathrm{ZADD}(\boldsymbol{P}, \boldsymbol{Q})$ yields for free

$$\boldsymbol{P}' = \left(X_1(X_1 - X_2)^2 : Y_1(X_1 - X_2)^3 : Z_3\right) = (W_1 : A_1 : Z_3) \sim \boldsymbol{P}$$

that is, $\mathrm{Z}(\boldsymbol{P}') = \mathrm{Z}(\boldsymbol{R})$

- Notation: $(\boldsymbol{R}, \boldsymbol{P}') = \mathrm{ZADDU}(\boldsymbol{P}, \boldsymbol{Q})$
- Cost of ZADDU: $\underline{5\mathrm{M} + 2\mathrm{S}}$

technicolor

Outline

1. Arithmetic on Elliptic Curves
   - Jacobian coordinates
   - Co-Z point addition
2. **Binary Scalar Multiplication Algorithms**
   - **Left-to-right methods**
   - **Right-to-left methods**
3. New Implementations
   - Binary ladders with co-Z trick
   - Point double-add operation
4. Discussion
   - Performance analysis
   - Security analysis
5. Conclusion

technicolor

Left-to-Right Methods

Algorithm 1 Left-to-right binary method

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{O}$; $\boldsymbol{R_1} \leftarrow \boldsymbol{P}$
2: **for** $i = n-1$ down to 0 **do**
3: $\quad \boldsymbol{R_0} \leftarrow 2\boldsymbol{R_0}$
4: $\quad$ **if** $(k_i = 1)$ **then** $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_1}$
5: **end for**
6: **return** $\boldsymbol{R_0}$

Algorithm 2 Montgomery ladder

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{O}$; $\boldsymbol{R_1} \leftarrow \boldsymbol{P}$
2: **for** $i = n-1$ down to 0 **do**
3: $\quad b \leftarrow k_i$; $\boldsymbol{R_{1-b}} \leftarrow \boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
4: $\quad \boldsymbol{R_b} \leftarrow 2\boldsymbol{R_b}$
5: **end for**
6: **return** $\boldsymbol{R_0}$

- Subject to SPA-type attacks
- Inserting dummy addition prevents SPA
  - subject to safe-error attacks

- Regular structure, no dummy operations
- Naturally resistant against SPA and safe-error attacks
- 2 registers

technicolor

Right-to-Left Methods

Algorithm 3 Right-to-left binary method

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{O}$; $\boldsymbol{R_1} \leftarrow \boldsymbol{P}$
2: **for** $i = 0$ to $n - 1$ **do**
3:     **if** $(k_i = 1)$ **then** $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_1}$
4:     $\boldsymbol{R_1} \leftarrow 2\boldsymbol{R_1}$
5: **end for**
6: **return** $\boldsymbol{R_0}$

- Idem left-to-right method

  (SPA-type attacks, safe-error attacks)

Algorithm 4 Joye's double-add

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{O}$; $\boldsymbol{R_1} \leftarrow \boldsymbol{P}$
2: **for** $i = 0$ to $n - 1$ **do**
3:     $b \leftarrow k_i$
4:     $\boldsymbol{R_{1-b}} \leftarrow 2\boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
5: **end for**
6: **return** $\boldsymbol{R_0}$

- Idem Montgomery ladder

  (regular structure, no dummy operations, 2 registers)

technicolor

Outline

1. Arithmetic on Elliptic Curves
   - Jacobian coordinates
   - Co-Z point addition
2. Binary Scalar Multiplication Algorithms
   - Left-to-right methods
   - Right-to-left methods
3. **New Implementations**
   - **Binary ladders with co-Z trick**
   - **Point double-add operation**
4. Discussion
   - Performance analysis
   - Security analysis
5. Conclusion

technicolor

Conjugate co-Z Point Addition (ZADDC)

- New co-Z point operation
  - using caching techniques

Conjugate co-Z point addition

From $-\boldsymbol{Q} = (X_2 : -Y_2 : Z_2)$, evaluation of $\boldsymbol{R} = \mathrm{ZADD}(\boldsymbol{P}, \boldsymbol{Q})$ allows one to get $\boldsymbol{S} := \boldsymbol{P} - \boldsymbol{Q} = (\overline{X_3}, \overline{Y_3}, Z_3)$ where

$$\overline{X_3} = (Y_1 + Y_2)^2 - W_1 - W_2, \quad \overline{Y_3} = (Y_1 + Y_2)(W_1 - \overline{X_3})$$

with an additional cost of $1\mathrm{M} + 1\mathrm{S}$

- Notation: $(\boldsymbol{P} + \boldsymbol{Q}, \boldsymbol{P} - \boldsymbol{Q}) = \mathrm{ZADDC}(\boldsymbol{P}, \boldsymbol{Q})$
- Total cost of ZADDC: $\underline{6\mathrm{M} + 3\mathrm{S}}$

technicolor

Left-to-Right Binary Ladder With co-Z Trick

Algorithm 5 Montgomery ladder with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_{n-1} = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

$\boldsymbol{R_0} \leftarrow \boldsymbol{O}$; $\boldsymbol{R_1} \leftarrow \boldsymbol{P}$
for $i = n - 1$ down to 0 **do**
    $b \leftarrow k_i$; $\boldsymbol{R_{1-b}} \leftarrow \boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
    $\boldsymbol{R_b} \leftarrow 2\boldsymbol{R_b}$
end for
return $\boldsymbol{R_0}$

Left-to-Right Binary Ladder With co-Z Trick

Algorithm 5 Montgomery ladder with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_{n-1} = 1$

Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_1}, \boldsymbol{R_0}) \leftarrow \mathrm{DBLU}(\boldsymbol{R_0})$
2: **for** $i = n-2$ down to 0 **do**
3: $\quad b \leftarrow k_i$; $\boldsymbol{R_{1-b}} \leftarrow \boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
4: $\quad \boldsymbol{R_b} \leftarrow 2\boldsymbol{R_b}$
5: **end for**
6: **return** $\boldsymbol{R_0}$

$(2\boldsymbol{P}, \boldsymbol{P'}) = \mathrm{DBLU}(\boldsymbol{P})$ where $\boldsymbol{P'} \sim \boldsymbol{P}$ and $\mathrm{Z}(\boldsymbol{P'}) = \mathrm{Z}(2\boldsymbol{P})$
Cost: $1\mathrm{M} + 5\mathrm{S}$

technicolor

Left-to-Right Binary Ladder With co-Z Trick

Algorithm 5 Montgomery ladder with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_{n-1} = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_1}, \boldsymbol{R_0}) \leftarrow \mathrm{DBLU}(\boldsymbol{R_0})$
2: **for** $i = n-2$ down to 0 **do**
3: $\quad b \leftarrow k_i$; $\boldsymbol{R_{1-b}} \leftarrow \boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
4: $\quad \boldsymbol{R_b} \leftarrow 2\boldsymbol{R_b}$
5: **end for**
6: **return** $\boldsymbol{R_0}$

$\boldsymbol{T} \leftarrow \boldsymbol{R_b} - \boldsymbol{R_{1-b}}$
$\boldsymbol{R_{1-b}} \leftarrow \boldsymbol{R_b} + \boldsymbol{R_{1-b}}$; $\boldsymbol{R_b} \leftarrow \boldsymbol{R_{1-b}} + \boldsymbol{T}$ $(= 2\boldsymbol{R_b})$

Left-to-Right Binary Ladder With co-Z Trick

Algorithm 5 Montgomery ladder with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_{n-1} = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_1}, \boldsymbol{R_0}) \leftarrow \mathrm{DBLU}(\boldsymbol{R_0})$
2: **for** $i = n-2$ down to 0 **do**
3: $\quad b \leftarrow k_i$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{ZADDC}(\boldsymbol{R_b}, \boldsymbol{R_{1-b}})$
4: $\quad (\boldsymbol{R_b}, \boldsymbol{R_{1-b}}) \leftarrow \mathrm{ZADDU}(\boldsymbol{R_{1-b}}, \boldsymbol{R_b})$
5: **end for**
6: **return** $\boldsymbol{R_0}$

$\boldsymbol{T} \leftarrow \boldsymbol{R_b} - \boldsymbol{R_{1-b}}$
$\boldsymbol{R_{1-b}} \leftarrow \boldsymbol{R_b} + \boldsymbol{R_{1-b}}$; $\boldsymbol{R_b} \leftarrow \boldsymbol{R_{1-b}} + \boldsymbol{T}$ $(= 2\boldsymbol{R_b})$

technicolor

Left-to-Right Binary Ladder With co-Z Trick

Algorithm 5 Montgomery ladder with co-Z formulæ

Input: $P \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_{n-1} = 1$
Output: $Q = kP$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_1}, \boldsymbol{R_0}) \leftarrow \mathrm{DBLU}(\boldsymbol{R_0})$
2: **for** $i = n-2$ down to 0 **do**
3: $\quad b \leftarrow k_i$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{ZADDC}(\boldsymbol{R_b}, \boldsymbol{R_{1-b}})$
4: $\quad (\boldsymbol{R_b}, \boldsymbol{R_{1-b}}) \leftarrow \mathrm{ZADDU}(\boldsymbol{R_{1-b}}, \boldsymbol{R_b})$
5: **end for**
6: **return** $\boldsymbol{R_0}$

- Cost per bit: $(6\mathrm{M} + 3\mathrm{S}) + (5\mathrm{M} + 2\mathrm{S}) = 11\mathrm{M} + 5\mathrm{S}$

Right-to-Left Binary Ladder With co-Z Trick

Algorithm 6 Joye's double-add with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_0 = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{O}$; $\boldsymbol{R_1} \leftarrow \boldsymbol{P}$
2: **for** $i = 0$ to $n-1$ **do**
3: $\quad b \leftarrow k_i$;
4: $\quad \boldsymbol{R_{1-b}} \leftarrow 2\boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
5: **end for**
6: **return** $\boldsymbol{R_0}$

technicolor

Right-to-Left Binary Ladder With co-Z Trick

Algorithm 6 Joye's double-add with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_0 = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $\boldsymbol{R_0} \leftarrow \boldsymbol{P}$; $\boldsymbol{R_1} \leftarrow \boldsymbol{P}$
2: **for** $i = 1$ to $n-1$ **do**
3: $\quad b \leftarrow k_i$;
4: $\quad \boldsymbol{R_{1-b}} \leftarrow 2\boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
5: **end for**
6: **return** $\boldsymbol{R_0}$

$\boldsymbol{R_0}$ and $\boldsymbol{R_1}$ now have the same Z-coordinate but are not different (!)
$\Longrightarrow$ start for-loop at $i = 2$

technicolor

Right-to-Left Binary Ladder With co-Z Trick

Algorithm 6 Joye's double-add with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_0 = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

```
b ← k_1; R_b ← P; (R_{1−b}, R_b) ← TPLU(R_b)
for i = 2 to n − 1 do
    b ← k_i;
    R_{1−b} ← 2R_{1−b} + R_b
end for
return R_0
```

$(3\boldsymbol{P}, \boldsymbol{P}') = \mathrm{TPLU}(\boldsymbol{P})$ where $\boldsymbol{P}' \sim \boldsymbol{P}$ and $\mathrm{Z}(\boldsymbol{P}') = \mathrm{Z}(3\boldsymbol{P})$
Cost: $6\mathrm{M} + 7\mathrm{S}$

Right-to-Left Binary Ladder With co-Z Trick

Algorithm 6 Joye's double-add with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_0 = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $b \leftarrow k_1$; $\boldsymbol{R_b} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{TPLU}(\boldsymbol{R_b})$
2: **for** $i = 2$ to $n-1$ **do**
3: $\quad b \leftarrow k_i$;
4: $\quad \boldsymbol{R_{1-b}} \leftarrow 2\boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
5: **end for**
6: **return** $\boldsymbol{R_0}$

Can be rewritten as $\boldsymbol{T} \leftarrow \boldsymbol{R_{1-b}} + \boldsymbol{R_b}$; $\quad \boldsymbol{R_{1-b}} \leftarrow \boldsymbol{T} + \boldsymbol{R_{1-b}}$

technicolor

Right-to-Left Binary Ladder With co-Z Trick

Algorithm 6 Joye's double-add with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_0 = 1$

Output: $\boldsymbol{Q} = k\boldsymbol{P}$

$b \leftarrow k_1$; $\boldsymbol{R_b} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{TPLU}(\boldsymbol{R_b})$
for $i = 2$ to $n-1$ **do**
  $b \leftarrow k_i$; $\boldsymbol{T} \leftarrow \boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
  $\boldsymbol{R_{1-b}} \leftarrow \boldsymbol{T} + \boldsymbol{R_{1-b}}$
end for
return $\boldsymbol{R_0}$

$(\boldsymbol{T}, \boldsymbol{R_{1-b}}) \leftarrow \mathrm{ZADDU}(\boldsymbol{R_{1-b}}, \boldsymbol{R_b})$; $(\boldsymbol{R_{1-b}}, \boldsymbol{T}) \leftarrow \mathrm{ZADDU}(\boldsymbol{T}, \boldsymbol{R_{1-b}})$

+ update of $\boldsymbol{R_b}$ (cost: 3M)

Right-to-Left Binary Ladder With co-Z Trick

Algorithm 6 Joye's double-add with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_0 = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

$b \leftarrow k_1$; $\boldsymbol{R_b} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{TPLU}(\boldsymbol{R_b})$
for $i = 2$ to $n-1$ **do**
    $b \leftarrow k_i$; $\boldsymbol{T} \leftarrow \boldsymbol{R_{1-b}} + \boldsymbol{R_b}$
    $\boldsymbol{R_{1-b}} \leftarrow \boldsymbol{T} + \boldsymbol{R_{1-b}}$
end for
return $\boldsymbol{R_0}$

$(\boldsymbol{T}, \boldsymbol{R_{1-b}}) \leftarrow \mathrm{ZADDU}(\boldsymbol{R_{1-b}}, \boldsymbol{R_b})$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{ZADDC}(\boldsymbol{T}, \boldsymbol{R_{1-b}})$
since $\boldsymbol{T} - \boldsymbol{R_{1-b}} = \boldsymbol{R_b}$

technicolor

Right-to-Left Binary Ladder With co-Z Trick

Algorithm 6 Joye's double-add with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_0 = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $b \leftarrow k_1$; $\boldsymbol{R_b} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{TPLU}(\boldsymbol{R_b})$
2: **for** $i = 2$ to $n-1$ **do**
3: $\quad b \leftarrow k_i$; $(\boldsymbol{R_b}, \boldsymbol{R_{1-b}}) \leftarrow \mathrm{ZADDU}(\boldsymbol{R_{1-b}}, \boldsymbol{R_b})$
4: $\quad (\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{ZADDC}(\boldsymbol{R_b}, \boldsymbol{R_{1-b}})$
5: **end for**
6: **return** $\boldsymbol{R_0}$

$(\boldsymbol{T}, \boldsymbol{R_{1-b}}) \leftarrow \mathrm{ZADDU}(\boldsymbol{R_{1-b}}, \boldsymbol{R_b})$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{ZADDC}(\boldsymbol{T}, \boldsymbol{R_{1-b}})$
since $\boldsymbol{T} - \boldsymbol{R_{1-b}} = \boldsymbol{R_b}$

technicolor

Right-to-Left Binary Ladder With co-Z Trick

Algorithm 6 Joye's double-add with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_0 = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

1: $b \leftarrow k_1$; $\boldsymbol{R_b} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{TPLU}(\boldsymbol{R_b})$
2: **for** $i = 2$ to $n-1$ **do**
3: $\quad b \leftarrow k_i$; $(\boldsymbol{R_b}, \boldsymbol{R_{1-b}}) \leftarrow \mathrm{ZADDU}(\boldsymbol{R_{1-b}}, \boldsymbol{R_b})$
4: $\quad (\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{ZADDC}(\boldsymbol{R_b}, \boldsymbol{R_{1-b}})$
5: **end for**
6: **return** $\boldsymbol{R_0}$

- Cost per bit: $(5\mathrm{M} + 2\mathrm{S}) + (6\mathrm{M} + 3\mathrm{S}) = \underline{11\mathrm{M} + 5\mathrm{S}}$

technicolor

Point Doubling-Addition

- Point doubling-addition evaluates: $\boldsymbol{R} \leftarrow 2\boldsymbol{P} + \boldsymbol{Q}$
  - $\boldsymbol{T} \leftarrow \boldsymbol{P} + \boldsymbol{Q}$ followed by $\begin{cases} \boldsymbol{R} \leftarrow \boldsymbol{T} + \boldsymbol{P} \\ \boldsymbol{Q} \leftarrow \boldsymbol{T} - \boldsymbol{P} \end{cases}$
  - $(\boldsymbol{T}, \boldsymbol{P}) \leftarrow \mathrm{ZADDU}(\boldsymbol{P}, \boldsymbol{Q})$; $(\boldsymbol{R}, \boldsymbol{Q}) \leftarrow \mathrm{ZADDC}(\boldsymbol{T}, \boldsymbol{P})$
  - cost: $11\mathrm{M} + 5\mathrm{S}$
- Combined operation

Co-Z point doubling-addition with update

$$(\boldsymbol{R}, \boldsymbol{Q}) \leftarrow \mathrm{ZDAU}(\boldsymbol{P}, \boldsymbol{Q})$$

  - trades 2M against 2S
  - cost: $\underline{9\mathrm{M} + 7\mathrm{S}}$

Application

Algorithm 7 Joye's double-add with co-Z formulæ

Input: $\boldsymbol{P} \in E(\mathbb{F}_q)$ and $k = (k_{n-1}, \ldots, k_0)_2 \in \mathbb{N}$ with $k_0 = 1$
Output: $\boldsymbol{Q} = k\boldsymbol{P}$

$b \leftarrow k_1$; $\boldsymbol{R_b} \leftarrow \boldsymbol{P}$; $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{TPLU}(\boldsymbol{R_b})$
for $i = 2$ to $n-1$ **do**
    $b \leftarrow k_i$
    $(\boldsymbol{R_{1-b}}, \boldsymbol{R_b}) \leftarrow \mathrm{ZDAU}(\boldsymbol{R_{1-b}}, \boldsymbol{R_b})$
end for
return $\boldsymbol{R_0}$

- Cost per bit: $9\mathrm{M} + 7\mathrm{S}$
- (Similar saving applies to Montgomery ladder)

technicolor

Outline

1 Arithmetic on Elliptic Curves
- Jacobian coordinates
- Co-Z point addition

2 Binary Scalar Multiplication Algorithms
- Left-to-right methods
- Right-to-left methods

3 New Implementations
- Binary ladders with co-Z trick
- Point double-add operation

4 **Discussion**
- **Performance analysis**
- **Security analysis**

5 Conclusion

technicolor

Performance: Addition Formulæ

Operation	Notation	Cost
Point addition:		
– general addition	ADD	$11M + 5S$
– co-Z addition	ZADD	$5M + 2S$
– co-Z addition with update	ZADDU	$\underline{5M + 2S}$
– general conjugate addition	ADDC	$12M + 6S$
– conjugate co-Z addition	ZADDC	$\underline{6M + 3S}$
Point doubling-addition:		
– general version	DA	$13M + 8S$
– mixed version	mDA	$11M + 7S$
– co-Z version with update	ZDAU	$\underline{9M + 7S}$

- Comparison
  - very efficient co-Z formulæ

technicolor

Performance: Scalar Multiplication

Algorithm	Operations	Cost per bit
Joye's double-add:		
— basic version	DA	13M + 8S
— co-Z version	ZDAU	9M + 7S
Montgomery ladder:		
— basic version	DBL and ADD	14M + 10S
— X-only version	XDBL and XADD	9M + 7S †
— co-Z version	ZDAU′	9M + 7S

† assuming that multiplications by a have negligible cost

- Comparison
  - co-Z versions are always faster
  - cost is independent of the curve parameters
- Latest news: cost reduced to 8M + 6S with new ZACAU′ op.

technicolor

Performance: Scalar Multiplication

Algorithm	Operations	Cost per bit
Joye's double-add:		
— basic version	DA	$13M + 8S$
— co-Z version	ZDAU	$9M + 7S$
Montgomery ladder:		
— basic version	DBL and ADD	$14M + 10S$
— X-only version	XDBL and XADD	$9M + 7S$ †
— co-Z version	ZACAU′	$8M + 6S$

† assuming that multiplications by a have negligible cost

- Comparison
  - co-Z versions are always faster
  - cost is independent of the curve parameters
- Latest news: cost reduced to $8M + 6S$ with new ZACAU′ op.

technicolor

Security Analysis

- Proposed co-Z implementations are built on highly regular scalar multiplication algorithms
  - inherit similar security features
  - naturally resistant against
    - SPA-type attacks
    - safe-error attacks
- Can be combined with existing DPA-type countermeasures
- Output complete point representation
  - possible to check redundant relations
    - e.g., output point belongs to the curve
  - useful feature against (regular) fault attacks

technicolor

Outline

1. **Arithmetic on Elliptic Curves**
   - Jacobian coordinates
   - Co-Z point addition
2. **Binary Scalar Multiplication Algorithms**
   - Left-to-right methods
   - Right-to-left methods
3. **New Implementations**
   - Binary ladders with co-Z trick
   - Point double-add operation
4. **Discussion**
   - Performance analysis
   - Security analysis
5. **Conclusion**

technicolor

Summary

- New strategies for evaluating scalar multiplications on elliptic curves using co-Z arithmetic
  - nicely combine with certain binary ladders
- Efficient co-Z conjugate point addition formula (as well as other companion co-Z formulæ)
  - require 7 or 8 registers
  - suitable for memory constrained devices

Full version available at http://eprint.iacr.org/2010/309

Acknowledgments

- We would like to thank
  - Jean-Luc Beuchat
  - Francisco Rodrìguez Henrìquez
  - Patrick Longa
  - Francesco Sica
  - Alexandre Venelli
  - An anonymous referee

technicolor

Questions?

technicolor