Complete addition formulas for prime order elliptic curves

Joost Renes[1] Craig Costello[2] Lejla Batina[1]

j.renes@cs.ru.nl

[1]Radboud University, Nijmegen, The Netherlands

[2]Microsoft Research, Redmond, USA

16 February 2016

About me

- PhD student (supervisor Lejla Batina)
- Digital Security Group
- Radboud University (Nijmegen, The Netherlands)
- (Academic) Interests:
  - Efficient and secure implementations of curve-based crypto
  - Side-channel analysis
  - (Hyper)elliptic-curve cryptography
  - Isogeny-based cryptography
- `http://www.cs.ru.nl/~jrenes/`

Outline

- Elliptic curve intro
- Complete formulas & comparison
- Background

Feel free to ask questions at any time!

Elliptic curves

$E(k)$: elliptic curve over a field k with $\mathrm{char}(k) \neq 2, 3$

Every elliptic curve can be written in short Weierstrass form

- Embedded in $\mathbb{P}^2$ as $E : Y^2Z = X^3 + aXZ^2 + bZ^3$
- The point $\mathcal{O} = (0 : 1 : 0)$ is called the point at infinity
- Affine points $(x : y : 1)$ given by $y^2 = x^3 + ax + b$

- The points on E form an abelian group under point addition $\oplus$ (with neutral element $\mathcal{O}$)
- Scalar multiplication $(k, P) \mapsto [k]P$ $(k \in \mathbb{Z}, P \in E)$

Elliptic curve cryptography (ECC)

Elliptic curve discrete logarithm problem (ECDLP)

Given two points $P, Q \in E$ such that $Q \in \langle P \rangle$. Find $k \in \mathbb{Z}$ such that $Q = [k]P$.

Commonly k is a secret, Q is public

- Key exchange: ECDH
- Signatures: ECDSA, EdDSA

Weierstrass model

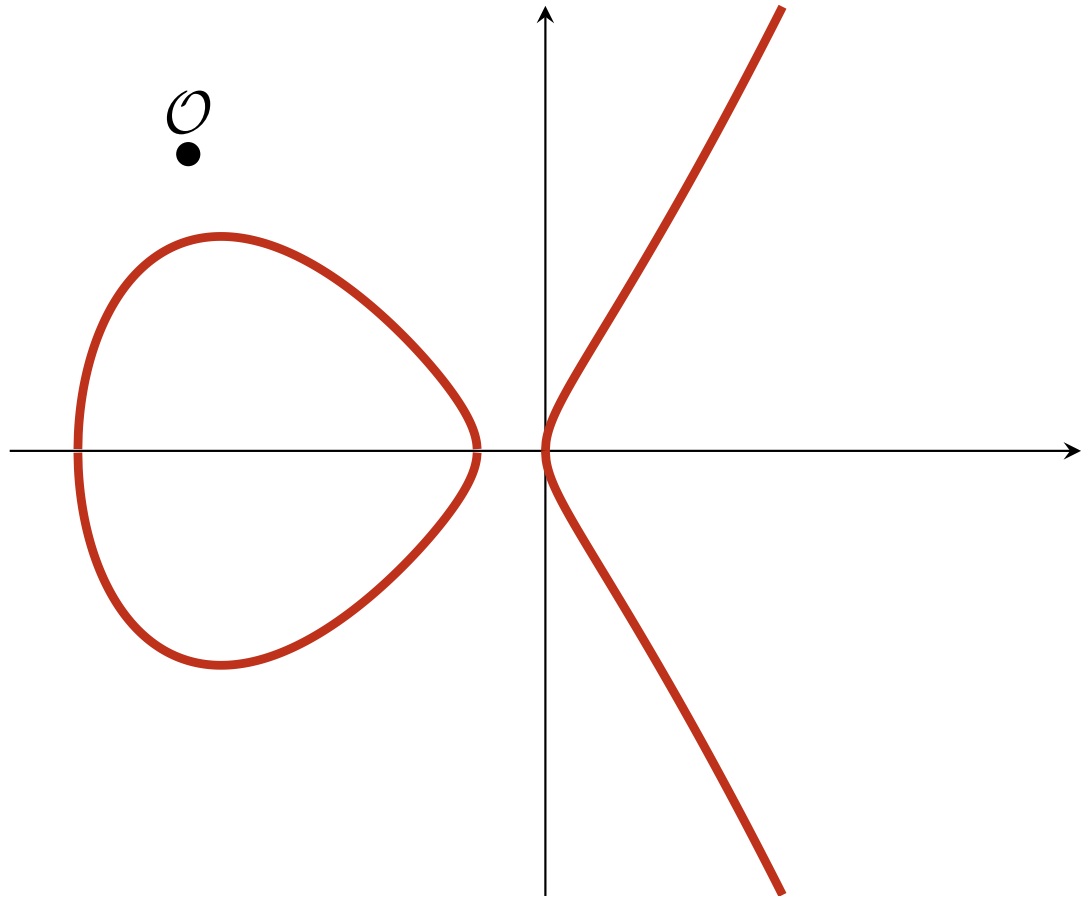

Figure: $E/\mathbb{R} : y^2 = x^3 + ax + b$

Addition

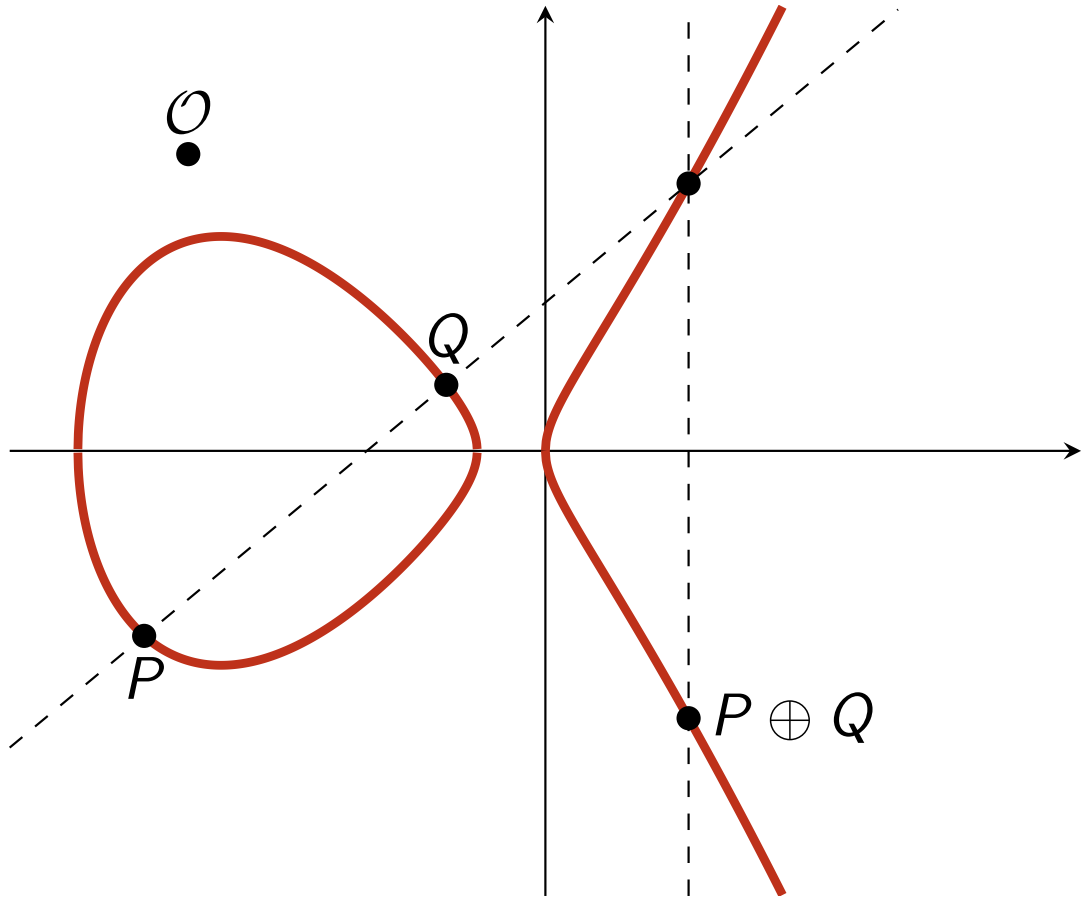

Figure: $E/\mathbb{R} : y^2 = x^3 + ax + b$

Addition

- if $P \neq \pm Q$
- if $P \neq \mathcal{O}$
- if $Q \neq \mathcal{O}$

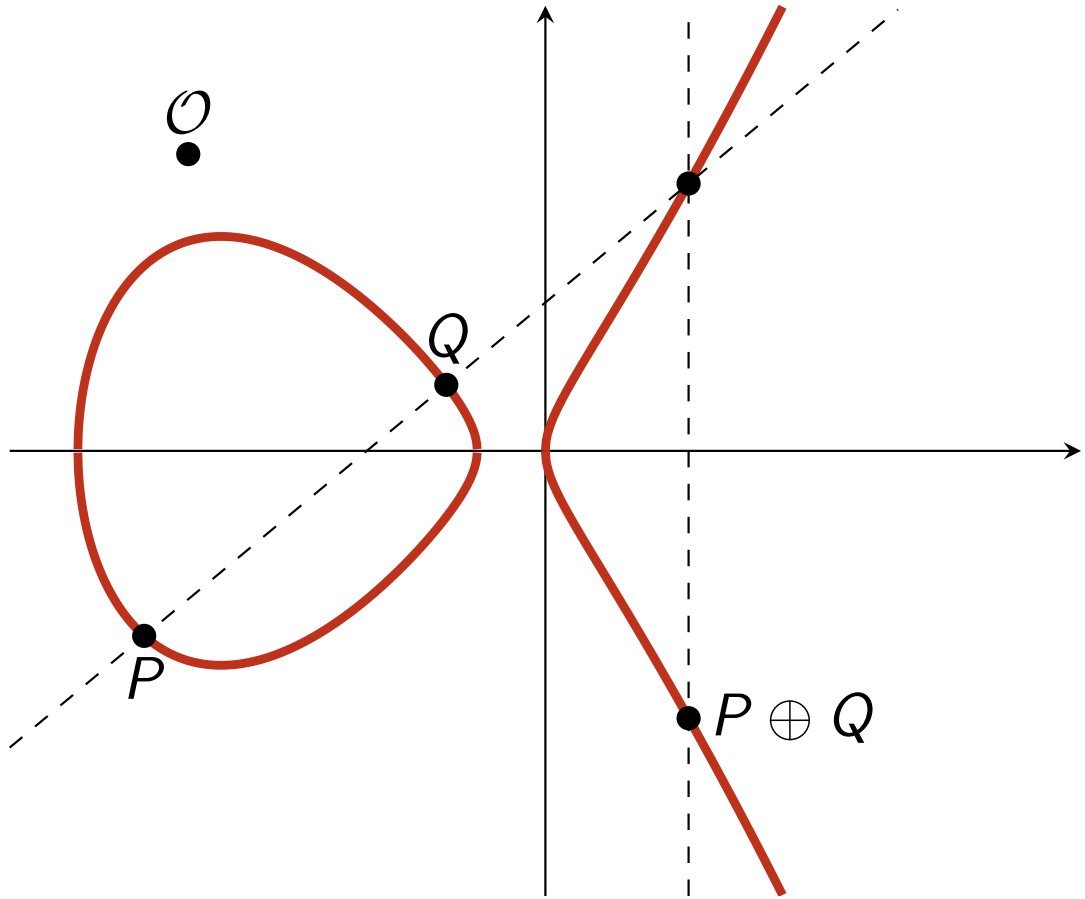

Figure: $E/\mathbb{R} : y^2 = x^3 + ax + b$

Doubling

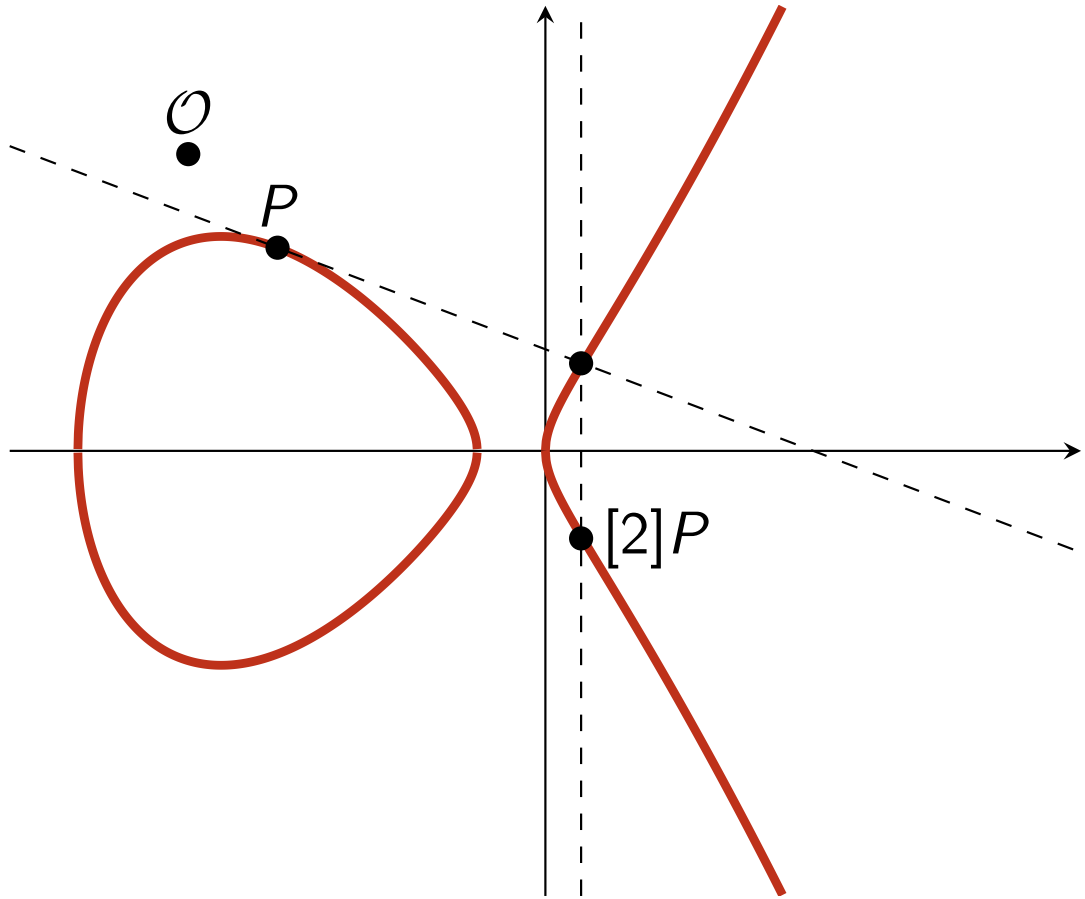

Figure: $E/\mathbb{R} : y^2 = x^3 + ax + b$

Doubling

- if $P \neq \mathcal{O}$

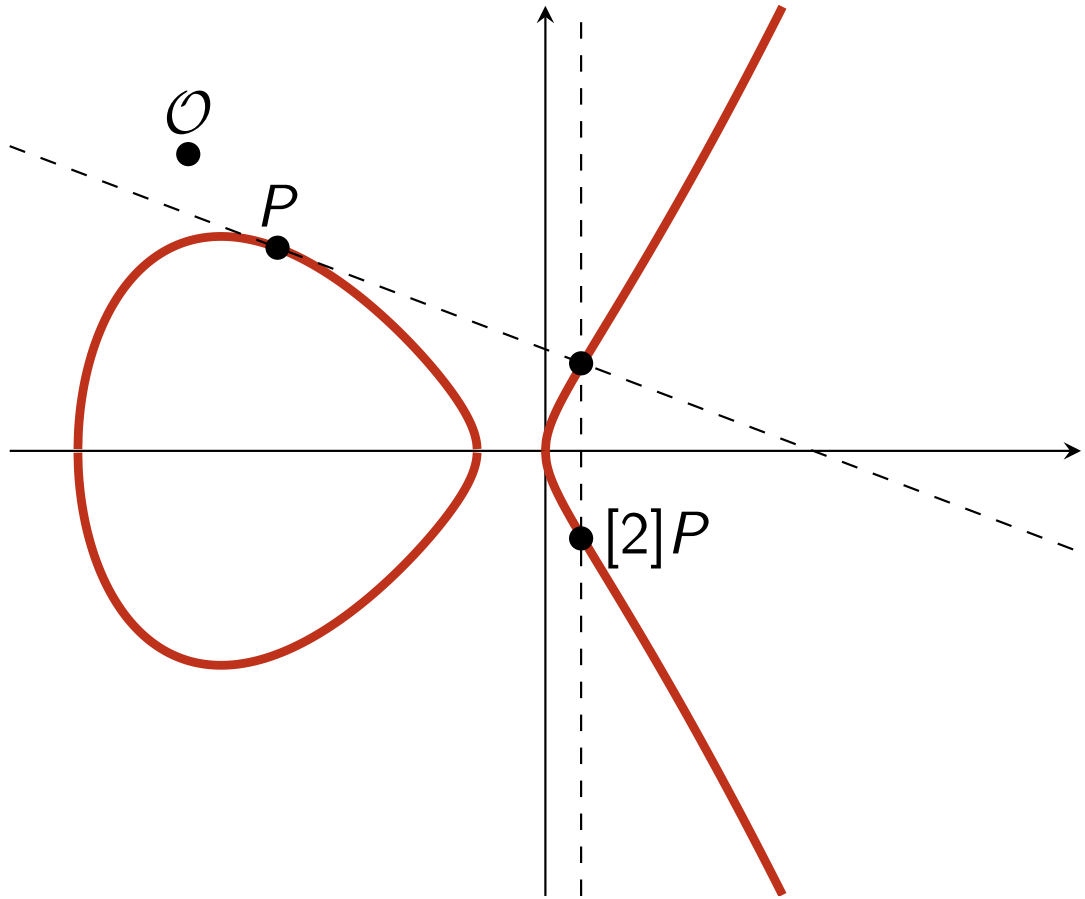

Figure: $E/\mathbb{R} : y^2 = x^3 + ax + b$

Implementation (Homogeneous addition)

$(X_1 : Y_1 : Z_1) \oplus (X_2 : Y_2 : Z_2) = (X_3 : Y_3 : Z_3)$, where:

$$X_3 = (X_2Z_1 - X_1Z_2)\Big[(Y_2Z_1 - Y_1Z_2)Z_1Z_2 - (X_2Z_1 - X_1Z_2)^3 - 2(X_2Z_1 - X_1Z_2)X_1Z_2\Big],$$

$$Y_3 = (Y_2Z_1 - Y_1Z_2)\Big[3(X_2Z_1 - X_1Z_2)X_1Z_2 - (Y_2Z_1 - Y_1Z_2)Z_1Z_2 + (X_2Z_1 - X_1Z_2)^3\Big] - (X_2Z_1 - X_1Z_2)^3 Y_1Z_2,$$

$$Z_3 = (X_2Z_1 - X_1Z_2)^3 Z_1Z_2.$$

But: $\left.\begin{array}{l} P = Q \\ P = \mathcal{O} \\ Q = \mathcal{O} \end{array}\right\} \implies X_3 = Y_3 = Z_3 = 0$ (**not in $\mathbb{P}^2$!**)

Implementation (Homogeneous doubling)

$[2](X : Y : Z) = (X_3 : Y_3 : Z_3)$, where

$$X_3 = 2\Big[(aZ^2 + 3X^2)^2 - 8XY^2Z\Big]YZ,$$
$$Y_3 = (aZ^2 + 3X^2)\Big[12XY^2Z - (aZ^2 + 3X^2)^2\Big] - 8Y^4Z^2,$$
$$Z_3 = 8Y^3Z^3.$$

But: $P = \mathcal{O} \implies X_3 = Y_3 = Z_3 = 0$ (**not in $\mathbb{P}^2$!**)

OpenSSL code example

```
int ec_GFp_simple_add(...)
{
    (...)

    if (a == b)
        return EC_POINT_dbl(group, r, a, ctx);
    if (EC_POINT_is_at_infinity(group, a))
        return EC_POINT_copy(r, b);
    if (EC_POINT_is_at_infinity(group, b))
        return EC_POINT_copy(r, a);

    (...)
}
```

OpenSSL code example

```
int ec_GFp_simple_add(...)
{
    (...)

    if (a == b)
        return EC_POINT_dbl(group, r, a, ctx);
    if (EC_POINT_is_at_infinity(group, a))
        return EC_POINT_copy(r, b);
    if (EC_POINT_is_at_infinity(group, b))
        return EC_POINT_copy(r, a);

    (...)
}
```

Exceptional cases

- Curves implemented using formulas with exceptional cases
- Handled by if-statements:
  - Code complexity
  - Bugs
  - Non-time-constant
  - Potential vulnerabilities

Standardized curves need to deal with this

- The example curves originally specified in the working drafts of ANSI, versions X9.62 and X9.63 [Acc99a; Acc99b].
- The five NIST prime curves specified in FIPS 186-4, i.e. P-192, P-224, P-256, P-384 and P-521.
- The seven curves specified in the German brainpool standard [ECC05], i.e., `brainpoolPXXXr1`, where $\texttt{XXX} \in \{160, 192, 224, 256, 320, 384, 512\}$.
- The eight curves specified by the UK-based company Certivox [Cer15], i.e., `ssc-XXX`, where $\texttt{XXX} \in \{160, 192, 224, 256, 288, 320, 384, 512\}$.
- The three curves specified (in addition to the above NIST prime curves) in the Certicom SEC 2 standard [Cer10]. This includes `secp256k1`, which is the curve used in the Bitcoin protocol.

A (partial) solution

- In 2007 Bernstein and Lange introduce *Edwards* curves
- Efficient exception-free addition formulas
- Problem: the curves have a cofactor
  $\Rightarrow$ Not possible for prime order curves
- Also the case for *twisted* Edwards and *Hessian* curves

Attempts for prime order curves

- For all NIST prime curves [BL09]: $26\mathbf{M} + 8\mathbf{S} + ...$
- *Unified* formulas [BJ02]: $11\mathbf{M} + 6\mathbf{S} + ...$
- Complete *system* of two addition laws [Bos+15]

Goal: *efficient* complete addition formulas for prime order curves

The result: complete addition formulas

Complete addition formulas for odd order subgroups

$(X_1 : Y_1 : Z_1) \oplus (X_2 : Y_2 : Z_2) = (X_3 : Y_3 : Z_3)$, where:

$$
\begin{aligned}
X_3 &= (X_1Y_2 + X_2Y_1)(Y_1Y_2 - a(X_1Z_2 + X_2Z_1) - 3bZ_1Z_2) \\
&\quad - (Y_1Z_2 + Y_2Z_1)(aX_1X_2 + 3b(X_1Z_2 + X_2Z_1) - a^2Z_1Z_2), \\
Y_3 &= (Y_1Y_2 + a(X_1Z_2 + X_2Z_1) + 3bZ_1Z_2)(Y_1Y_2 - a(X_1Z_2 + X_2Z_1) - 3bZ_1Z_2) \\
&\quad + (3X_1X_2 + aZ_1Z_2)(aX_1X_2 + 3b(X_1Z_2 + X_2Z_1) - a^2Z_1Z_2), \\
Z_3 &= (Y_1Z_2 + Y_2Z_1)(Y_1Y_2 + a(X_1Z_2 + X_2Z_1) + 3bZ_1Z_2) \\
&\quad + (X_1Y_2 + X_2Y_1)(3X_1X_2 + aZ_1Z_2).
\end{aligned}
$$

In particular this would work in any prime order group, including those on Edwards and Hessian curves

Operation count

any a:
$$\begin{cases} 12\mathbf{M} + 3\mathbf{m_a} + 2\mathbf{m_{3b}} + 23\mathbf{a} & P \oplus Q \\ 8\mathbf{M} + 3\mathbf{S} + 3\mathbf{m_a} + 2\mathbf{m_{3b}} + 15\mathbf{a} & [2]P \end{cases}$$

$a = -3$:
$$\begin{cases} 12\mathbf{M} + 2\mathbf{m_b} + 29\mathbf{a} & P \oplus Q \\ 8\mathbf{M} + 3\mathbf{S} + 2\mathbf{m_b} + 21\mathbf{a} & [2]P \end{cases}$$

$a = 0$:
$$\begin{cases} 12\mathbf{M} + 2\mathbf{m_{3b}} + 19\mathbf{a} & P \oplus Q \\ 6\mathbf{M} + 2\mathbf{S} + 1\mathbf{m_{3b}} + 9\mathbf{a} & [2]P \end{cases}$$

A comparison (any a)

- **This work** (addition): $12\mathbf{M} + 3\mathbf{m_a} + 2\mathbf{m_{3b}} + 23\mathbf{a}$
- **This work** (doubling): $8\mathbf{M} + 3\mathbf{S} + 3\mathbf{m_a} + 2\mathbf{m_{3b}} + 15\mathbf{a}$
- For all NIST prime curves [BL09]: $26\mathbf{M} + 8\mathbf{S} + ...$
- *Unified* formulas [BJ02]: $11\mathbf{M} + 6\mathbf{S} + ...$
- Jacobian coordinates addition: $12\mathbf{M} + 4\mathbf{S} + 7\mathbf{a}$
- Jacobian coordinates doubling: $3\mathbf{M} + 6\mathbf{S} + 1\mathbf{m_a} + 13\mathbf{a}$

A comparison (any a)

- **This work** (addition): $12\mathbf{M} + 3\mathbf{m_a} + 2\mathbf{m_{3b}} + 23\mathbf{a}$
- **This work** (doubling): $8\mathbf{M} + 3\mathbf{S} + 3\mathbf{m_a} + 2\mathbf{m_{3b}} + 15\mathbf{a}$
- For all NIST prime curves [BL09]: $26\mathbf{M} + 8\mathbf{S} + ...$
- *Unified* formulas [BJ02]: $11\mathbf{M} + 6\mathbf{S} + ...$
- Jacobian coordinates addition: $12\mathbf{M} + 4\mathbf{S} + 7\mathbf{a}$
- Jacobian coordinates doubling: $3\mathbf{M} + 6\mathbf{S} + 1\mathbf{m_a} + 13\mathbf{a}$

A comparison (any a)

- **This work** (addition): $12\mathbf{M} + 3\mathbf{m_a} + 2\mathbf{m_{3b}} + 23\mathbf{a}$
- **This work** (doubling): $8\mathbf{M} + 3\mathbf{S} + 3\mathbf{m_a} + 2\mathbf{m_{3b}} + 15\mathbf{a}$
- For all NIST prime curves [BL09]: $26\mathbf{M} + 8\mathbf{S} + ...$
- *Unified* formulas [BJ02]: $11\mathbf{M} + 6\mathbf{S} + ...$
- Jacobian coordinates addition: $12\mathbf{M} + 4\mathbf{S} + 7\mathbf{a}$
- Jacobian coordinates doubling: $3\mathbf{M} + 6\mathbf{S} + 1\mathbf{m_a} + 13\mathbf{a}$

A software comparison: OpenSSL

NIST curve	no. of ECDH operations (per 10s)		factor slowdown
	complete	incomplete	
P-192	35274	47431	1.34x
P-224	24810	34313	1.38x
P-256	21853	30158	1.38x
P-384	10109	14252	1.41x
P-521	4580	6634	1.44x

Table: Number of ECDH operations in 10 seconds for the OpenSSL implementation of the five NIST prime curves. Timings were obtained by running the "`openssl speed ecdhpXXX`" command on an Intel Core i5-5300 CPU @ 2.30GHz, averaged over 100 trials of 10s each.

A hardware comparison: FPGA implementation [MRB16]

For all prime order curves over prime fields of up to 522 bits

- A single set of formulas
- Built on top of Montgomery modular multiplier
  - Additions very cheap compared to multiplications
  - No distinction between multiplications and squarings
- Benefit a lot from parallelizing formulas

Parallelizing

n	*Cost*	*Area* $\times$ *Time*
1	$17\mathbf{M} + 23\mathbf{a}$	$17\mathbf{M} + 23\mathbf{a}$
2	$9\mathbf{M_2} + 12\mathbf{a_2}$	$18\mathbf{M} + 24\mathbf{a}$
3	$6\mathbf{M_3} + 8\mathbf{a_3}$	$18\mathbf{M} + 24\mathbf{a}$
4	$5\mathbf{M_4} + 7\mathbf{a_4}$	$20\mathbf{M} + 28\mathbf{a}$
5	$4\mathbf{M_5} + 6\mathbf{a_5}$	$20\mathbf{M} + 30\mathbf{a}$
6	$3\mathbf{M_6} + 6\mathbf{a_6}$	$18\mathbf{M} + 36\mathbf{a}$

Parallelizing

n	*Cost*	*Area* $\times$ *Time*
1	$17\mathbf{M} + 23\mathbf{a}$	$17\mathbf{M} + 23\mathbf{a}$
2	$9\mathbf{M_2} + 12\mathbf{a_2}$	$18\mathbf{M} + 24\mathbf{a}$
3	$6\mathbf{M_3} + 8\mathbf{a_3}$	$18\mathbf{M} + 24\mathbf{a}$
4	$5\mathbf{M_4} + 7\mathbf{a_4}$	$20\mathbf{M} + 28\mathbf{a}$
5	$4\mathbf{M_5} + 6\mathbf{a_5}$	$20\mathbf{M} + 30\mathbf{a}$
6	$3\mathbf{M_6} + 6\mathbf{a_6}$	$18\mathbf{M} + 36\mathbf{a}$

Algorithm

	MM0	MM1	MM2
1:	$t_0 \leftarrow X_1 \cdot X_2;$	$t_1 \leftarrow Y_1 \cdot Y_2;$	$t_2 \leftarrow Z_1 \cdot Z_2;$
2:	$t_3 \leftarrow X_1 + Y_1;$	$t_4 \leftarrow X_2 + Y_2;$	$t_5 \leftarrow Y_1 + Z_1;$
3:	$t_6 \leftarrow Y_2 + Z_2;$	$t_7 \leftarrow X_1 + Z_1;$	$t_8 \leftarrow X_2 + Z_2;$
4:	$t_9 \leftarrow t_3 \cdot t_4;$	$t_{10} \leftarrow t_5 \cdot t_6;$	$t_{11} \leftarrow t_7 \cdot t_8;$
5:	$t_3 \leftarrow t_0 + t_1;$	$t_4 \leftarrow t_1 + t_2;$	$t_5 \leftarrow t_0 + t_2;$
6:	$t_6 \leftarrow b_3 \cdot t_2;$	$t_8 \leftarrow a \cdot t_2;$	
7:	$t_2 \leftarrow t_9 - t_3;$	$t_9 \leftarrow t_0 + t_0;$	$t_3 \leftarrow t_{10} - t_4;$
8:	$t_{10} \leftarrow t_9 + t_0;$	$t_4 \leftarrow t_{11} - t_5;$	$t_7 \leftarrow t_0 - t_8;$
9:	$t_0 \leftarrow a \cdot t_4;$	$t_5 \leftarrow b_3 \cdot t_4;$	$t_9 \leftarrow a \cdot t_7;$
10:	$t_4 \leftarrow t_0 + t_6;$	$t_7 \leftarrow t_5 + t_9;$	$t_0 \leftarrow t_8 + t_{10};$
11:	$t_5 \leftarrow t_1 - t_4;$	$t_6 \leftarrow t_1 + t_4;$	
12:	$t_1 \leftarrow t_5 \cdot t_6;$	$t_4 \leftarrow t_0 \cdot t_7;$	$t_8 \leftarrow t_3 \cdot t_7;$
13:	$t_9 \leftarrow t_2 \cdot t_5;$	$t_{10} \leftarrow t_3 \cdot t_6;$	$t_{11} \leftarrow t_0 \cdot t_2;$
14:	$X_3 \leftarrow t_9 - t_8;$	$Y_3 \leftarrow t_1 + t_4;$	$Z_3 \leftarrow t_{10} + t_{11};$

Algorithm

	MM0	MM1	MM2
1:	$t_0 \leftarrow X_1 \cdot X_2;$	$t_1 \leftarrow Y_1 \cdot Y_2;$	$t_2 \leftarrow Z_1 \cdot Z_2;$
2:	$t_3 \leftarrow X_1 + Y_1;$	$t_4 \leftarrow X_2 + Y_2;$	$t_5 \leftarrow Y_1 + Z_1;$
3:	$t_6 \leftarrow Y_2 + Z_2;$	$t_7 \leftarrow X_1 + Z_1;$	$t_8 \leftarrow X_2 + Z_2;$
4:	$t_9 \leftarrow t_3 \cdot t_4;$	$t_{10} \leftarrow t_5 \cdot t_6;$	$t_{11} \leftarrow t_7 \cdot t_8;$
5:	$t_3 \leftarrow t_0 + t_1;$	$t_4 \leftarrow t_1 + t_2;$	$t_5 \leftarrow t_0 + t_2;$
6:	$t_6 \leftarrow b_3 \cdot t_2;$	$t_8 \leftarrow a \cdot t_2;$	
7:	$t_2 \leftarrow t_9 - t_3;$	$t_9 \leftarrow t_0 + t_0;$	$t_3 \leftarrow t_{10} - t_4;$
8:	$t_{10} \leftarrow t_9 + t_0;$	$t_4 \leftarrow t_{11} - t_5;$	$t_7 \leftarrow t_0 - t_8;$
9:	$t_0 \leftarrow a \cdot t_4;$	$t_5 \leftarrow b_3 \cdot t_4;$	$t_9 \leftarrow a \cdot t_7;$
10:	$t_4 \leftarrow t_0 + t_6;$	$t_7 \leftarrow t_5 + t_9;$	$t_0 \leftarrow t_8 + t_{10};$
11:	$t_5 \leftarrow t_1 - t_4;$	$t_6 \leftarrow t_1 + t_4;$	
12:	$t_1 \leftarrow t_5 \cdot t_6;$	$t_4 \leftarrow t_0 \cdot t_7;$	$t_8 \leftarrow t_3 \cdot t_7;$
13:	$t_9 \leftarrow t_2 \cdot t_5;$	$t_{10} \leftarrow t_3 \cdot t_6;$	$t_{11} \leftarrow t_0 \cdot t_2;$
14:	$X_3 \leftarrow t_9 - t_8;$	$Y_3 \leftarrow t_1 + t_4;$	$Z_3 \leftarrow t_{10} + t_{11};$

Algorithm

	MM0	MM1	MM2
1:	$t_0 \leftarrow X_1 \cdot X_2;$	$t_1 \leftarrow Y_1 \cdot Y_2;$	$t_2 \leftarrow Z_1 \cdot Z_2;$
2:	$t_3 \leftarrow X_1 + Y_1;$	$t_4 \leftarrow X_2 + Y_2;$	$t_5 \leftarrow Y_1 + Z_1;$
3:	$t_6 \leftarrow Y_2 + Z_2;$	$t_7 \leftarrow X_1 + Z_1;$	$t_8 \leftarrow X_2 + Z_2;$
4:	$t_9 \leftarrow t_3 \cdot t_4;$	$t_{10} \leftarrow t_5 \cdot t_6;$	$t_{11} \leftarrow t_7 \cdot t_8;$
5:	$t_3 \leftarrow t_0 + t_1;$	$t_4 \leftarrow t_1 + t_2;$	$t_5 \leftarrow t_0 + t_2;$
6:	$t_6 \leftarrow b_3 \cdot t_2;$	$t_8 \leftarrow a \cdot t_2;$	
7:	$t_2 \leftarrow t_9 - t_3;$	$t_9 \leftarrow t_0 + t_0;$	$t_3 \leftarrow t_{10} - t_4;$
8:	$t_{10} \leftarrow t_9 + t_0;$	$t_4 \leftarrow t_{11} - t_5;$	$t_7 \leftarrow t_0 - t_8;$
9:	$t_0 \leftarrow a \cdot t_4;$	$t_5 \leftarrow b_3 \cdot t_4;$	$t_9 \leftarrow a \cdot t_7;$
10:	$t_4 \leftarrow t_0 + t_6;$	$t_7 \leftarrow t_5 + t_9;$	$t_0 \leftarrow t_8 + t_{10};$
11:	$t_5 \leftarrow t_1 - t_4;$	$t_6 \leftarrow t_1 + t_4;$	
12:	$t_1 \leftarrow t_5 \cdot t_6;$	$t_4 \leftarrow t_0 \cdot t_7;$	$t_8 \leftarrow t_3 \cdot t_7;$
13:	$t_9 \leftarrow t_2 \cdot t_5;$	$t_{10} \leftarrow t_3 \cdot t_6;$	$t_{11} \leftarrow t_0 \cdot t_2;$
14:	$X_3 \leftarrow t_9 - t_8;$	$Y_3 \leftarrow t_1 + t_4;$	$Z_3 \leftarrow t_{10} + t_{11};$

Algorithm

	MM0	MM1	MM2
1:	$t_0 \leftarrow X_1 \cdot X_2;$	$t_1 \leftarrow Y_1 \cdot Y_2;$	$t_2 \leftarrow Z_1 \cdot Z_2;$
2:	$t_3 \leftarrow X_1 + Y_1;$	$t_4 \leftarrow X_2 + Y_2;$	$t_5 \leftarrow Y_1 + Z_1;$
3:	$t_6 \leftarrow Y_2 + Z_2;$	$t_7 \leftarrow X_1 + Z_1;$	$t_8 \leftarrow X_2 + Z_2;$
4:	$t_9 \leftarrow t_3 \cdot t_4;$	$t_{10} \leftarrow t_5 \cdot t_6;$	$t_{11} \leftarrow t_7 \cdot t_8;$
5:	$t_3 \leftarrow t_0 + t_1;$	$t_4 \leftarrow t_1 + t_2;$	$t_5 \leftarrow t_0 + t_2;$
6:	$t_6 \leftarrow b_3 \cdot t_2;$	$t_8 \leftarrow a \cdot t_2;$	6xADD
7:	$t_0 \leftarrow a \cdot t_4;$	$t_5 \leftarrow b_3 \cdot t_4;$	$t_9 \leftarrow a \cdot t_7;$
8:	$t_4 \leftarrow t_0 + t_6;$	$t_7 \leftarrow t_5 + t_9;$	$t_0 \leftarrow t_8 + t_{10};$
9:	$t_5 \leftarrow t_1 - t_4;$	$t_6 \leftarrow t_1 + t_4;$	
10:	$t_1 \leftarrow t_5 \cdot t_6;$	$t_4 \leftarrow t_0 \cdot t_7;$	$t_8 \leftarrow t_3 \cdot t_7;$
11:	$t_9 \leftarrow t_2 \cdot t_5;$	$t_{10} \leftarrow t_3 \cdot t_6;$	$t_{11} \leftarrow t_0 \cdot t_2;$
12:	$X_3 \leftarrow t_9 - t_8;$	$Y_3 \leftarrow t_1 + t_4;$	$Z_3 \leftarrow t_{10} + t_{11};$

A hardware comparison

Work	FPGA	LUT	FF	Freq. (MHz)	Scalar Mult. (ms)
For all prime fields and prime order short Weierstrass curves					
Our	IGLOO 2[4]	2967	1159	165	8.61
For NIST curves [Nat13] only					
[VGM11]	SmartFusion[4]	3690	3690	109	19.3
[VGM11]	Virtex II Pro[4]	1546	1546	210	10.02
[VGM11]	Virtex II Pro[4]	2316	2316	210	4.52
[PMG14]	Virtex 5[6]	7656	7656	210	3.95
[RDM15]	Spartan 6[6]	193	35	156.25	12.20
[LK15]	Virtex 4[4]	12435	3545	182	5.46
[AR14]	Virtex 6[6]	32.9k	89.6k	100	0.40
[GP08]	Virtex 4[4]	2589	2028	490	0.62
For only Edwards or Twisted Edwards curves					
[SG14]	Zynq[6]	2783	3592	200	0.32
For only specific field size, but works with any prime					
[Vli+10]	Virtex II Pro[4]	3664	3664	108.2	29.83
[Vli+10]	Virtex II Pro[4]	4170	4170	68.17	15.76
[Gui10]	Stratix II[4]	18354	18354	157.2	0.68
[MMM06]	Virtex II Pro[4]	31510	31510	39.46	3.86
[Ma+14]	Virtex 4[4]	5740	4876	250	0.44
[Bal+12]	Virtex 5[6]	7822	5780	81.71	4.04

A hardware comparison

Work	FPGA	LUT	FF	Freq. (MHz)	Scalar Mult. (ms)
For all prime fields and prime order short Weierstrass curves					
Our	IGLOO 2[4]	2967	1159	165	8.61
For NIST curves [Nat13] only					
[VGM11]	SmartFusion[4]	3690	3690	109	19.3
[VGM11]	Virtex II Pro[4]	1546	1546	210	10.02
[VGM11]	Virtex II Pro[4]	2316	2316	210	4.52
[PMG14]	Virtex 5[6]	7656	7656	210	3.95
[RDM15]	Spartan 6[6]	193	35	156.25	12.20
[LK15]	Virtex 4[4]	12435	3545	182	5.46
[AR14]	Virtex 6[6]	32.9k	89.6k	100	0.40
[GP08]	Virtex 4[4]	2589	2028	490	0.62
For only Edwards or Twisted Edwards curves					
[SG14]	Zynq[6]	2783	3592	200	0.32
For only specific field size, but works with any prime					
[Vli+10]	Virtex II Pro[4]	3664	3664	108.2	29.83
[Vli+10]	Virtex II Pro[4]	4170	4170	68.17	15.76
[Gui10]	Stratix II[4]	18354	18354	157.2	0.68
[MMM06]	Virtex II Pro[4]	31510	31510	39.46	3.86
[Ma+14]	Virtex 4[4]	5740	4876	250	0.44
[Bal+12]	Virtex 5[6]	7822	5780	81.71	4.04

A hardware comparison

Work	FPGA	LUT	FF	Freq. (MHz)	Scalar Mult. (ms)
For all prime fields and prime order short Weierstrass curves					
Our	IGLOO 2[4]	2967	1159	165	8.61
For NIST curves [Nat13] only					
[VGM11]	SmartFusion[4]	3690	3690	109	19.3
[VGM11]	Virtex II Pro[4]	1546	1546	210	10.02
[VGM11]	Virtex II Pro[4]	2316	2316	210	4.52
[PMG14]	Virtex 5[6]	7656	7656	210	3.95
[RDM15]	Spartan 6[6]	193	35	156.25	12.20
[LK15]	Virtex 4[4]	12435	3545	182	5.46
[AR14]	Virtex 6[6]	32.9k	89.6k	100	0.40
[GP08]	Virtex 4[4]	2589	2028	490	0.62
For only Edwards or Twisted Edwards curves					
[SG14]	Zynq[6]	2783	3592	200	0.32
For only specific field size, but works with any prime					
[Vli+10]	Virtex II Pro[4]	3664	3664	108.2	29.83
[Vli+10]	Virtex II Pro[4]	4170	4170	68.17	15.76
[Gui10]	Stratix II[4]	18354	18354	157.2	0.68
[MMM06]	Virtex II Pro[4]	31510	31510	39.46	3.86
[Ma+14]	Virtex 4[4]	5740	4876	250	0.44
[Bal+12]	Virtex 5[6]	7822	5780	81.71	4.04

Summarized

Complete addition formulas for odd order subgroups

- "Efficiently" computable
- Reduced code complexity
- Backwards compatibility with standardized curves
- Compatibility with cofactor curves
  - For constrained devices

Note: It is **not** a solution to all attacks

Background

- Lange and Ruppert consider complete systems of addition laws on abelian varieties [LR85]
- Bosma and Lenstra look at complete systems of addition laws for elliptic curves [BL95]
- Arène, Kohel and Ritzenthaler [AKR12] generalize results of [BL95] again to abelian varieties

Addition formulas

Addition formulas of bidegree (μ, ν) [BL95]

Tuple of polynomials (X_3, Y_3, Z_3) s.t. for all $(P, Q) \in E \times E$ either

1. $(X_3(P, Q) : Y_3(P, Q) : Z_3(P, Q)) = P \oplus Q$, or
2. $X_3(P, Q) = Y_3(P, Q) = Z_3(P, Q) = 0$,

where X_3, Y_3, Z_3 are homogeneous of degree μ resp. ν in the coordinates of P resp. Q

- If 2 holds for a pair (P, Q), it is called exceptional
- If 2 holds for none of the pairs (P, Q), the addition formulas (X_3, Y_3, Z_3) are called complete

Bosma-Lenstra Theorem 1

[BL95, Theorem 1]

The smallest cardinality of a complete system of addition laws on E equals two, and if two addition laws form a complete system then each of them has bidegree $(2, 2)$

Bosma-Lenstra Theorem 1

[BL95, Theorem 1]

The smallest cardinality of a complete system of addition laws on E equals two, and if two addition laws form a complete system then each of them has bidegree $(2, 2)$

- Only over algebraically closed field!
- Cryptographic interest in $k = \mathbb{F}_q$, so need to make sure the exceptional pairs lie in extension fields

Bosma-Lenstra Theorem 2

[BL95, Theorem 2]

There is a bijection between $\mathbb{P}^2(k)$ and the set of equivalence classes of non-zero addition laws of bidegree $(2, 2)$ on E that has the following property. [...]

Bosma-Lenstra Theorem 2

[BL95, Theorem 2]

There is a bijection between $\mathbb{P}^2(k)$ and the set of equivalence classes of non-zero addition laws of bidegree $(2,2)$ on E that has the following property. [...]

Given $(a : b : c) \in \mathbb{P}^2$, we define a line $L : aX + bY + cZ = 0$.

$$\text{Addition law complete} \iff L \cap E = \emptyset$$

B-L equivalence

- $L_1 : Y - 2Z = 0$

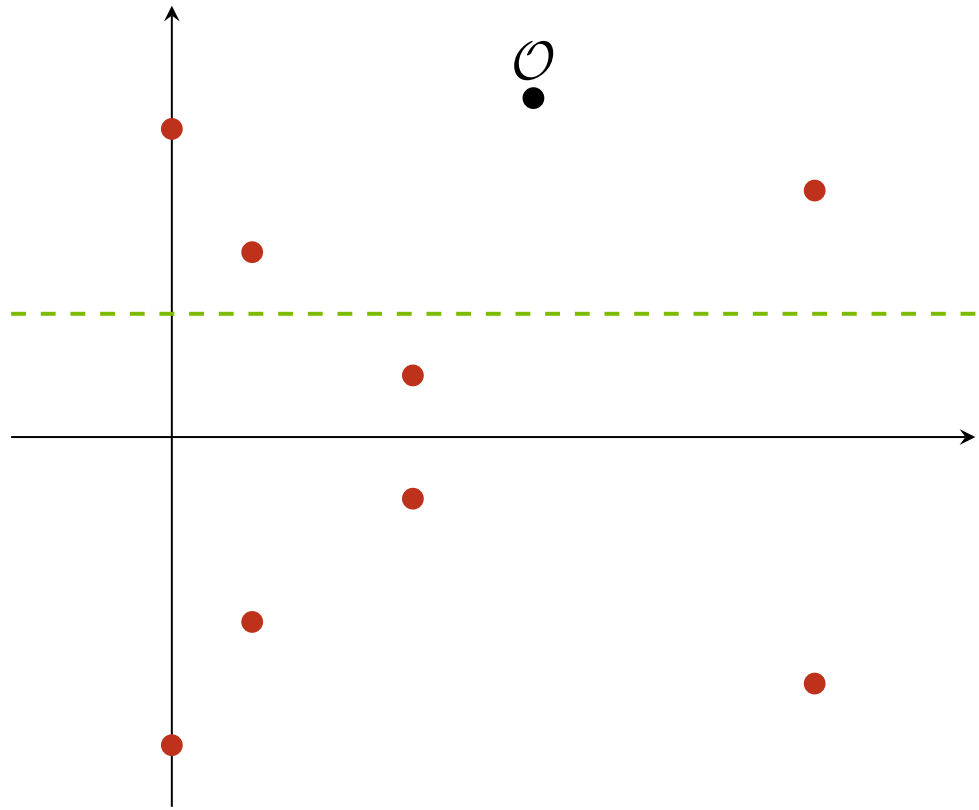

Figure: $E/\mathbb{F}_{11} : y^2 = x^3 + 5x + 3$

B-L equivalence

- $L_1 : Y - 2Z = 0$
- $L_2 : \frac{3}{5}X - Y - \frac{4}{5}Z = 0$

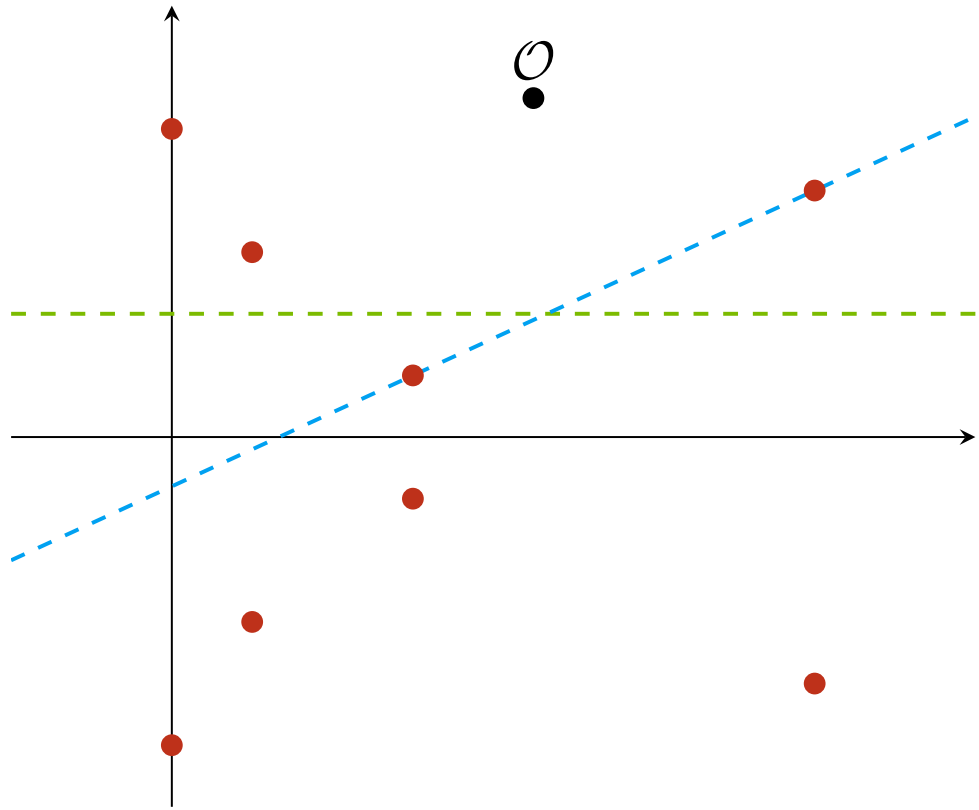

Figure: $E/\mathbb{F}_{11} : y^2 = x^3 + 5x + 3$

B-L equivalence

- $L_1 : Y - 2Z = 0$
- $L_2 : \frac{3}{5}X - Y - \frac{4}{5}Z = 0$
- $L_3 : X - Z = 0$

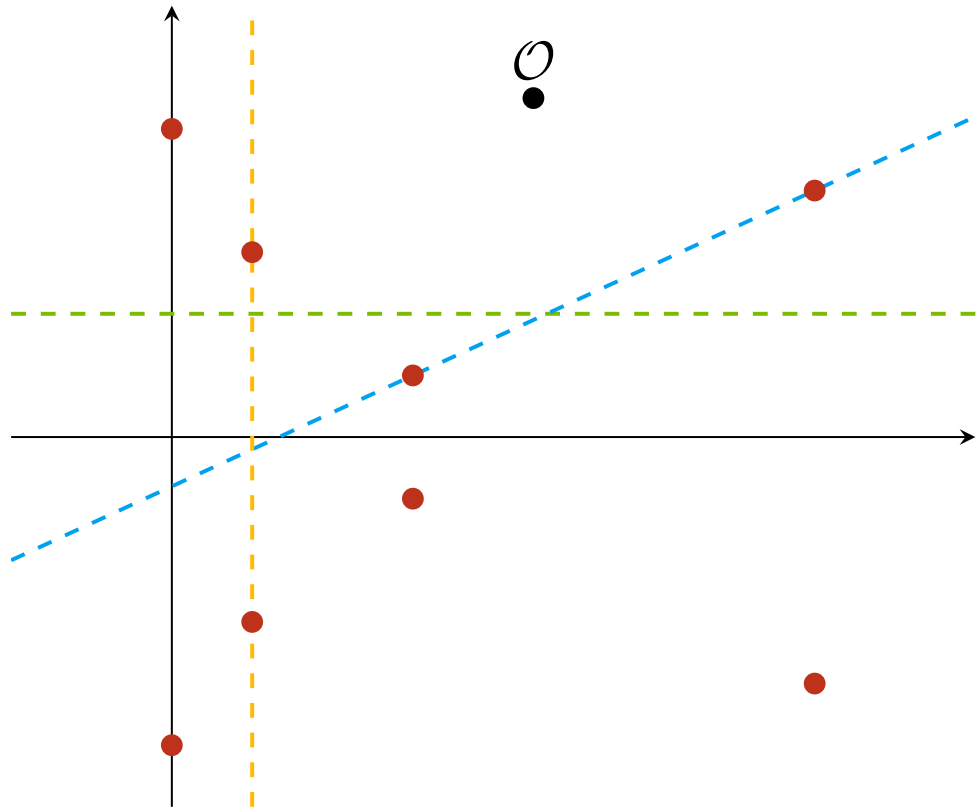

Figure: $E/\mathbb{F}_{11} : y^2 = x^3 + 5x + 3$

Choosing a basis

- $L_1 : X = 0$
- $L_2 : Y = 0$
- $L_3 : Z = 0$

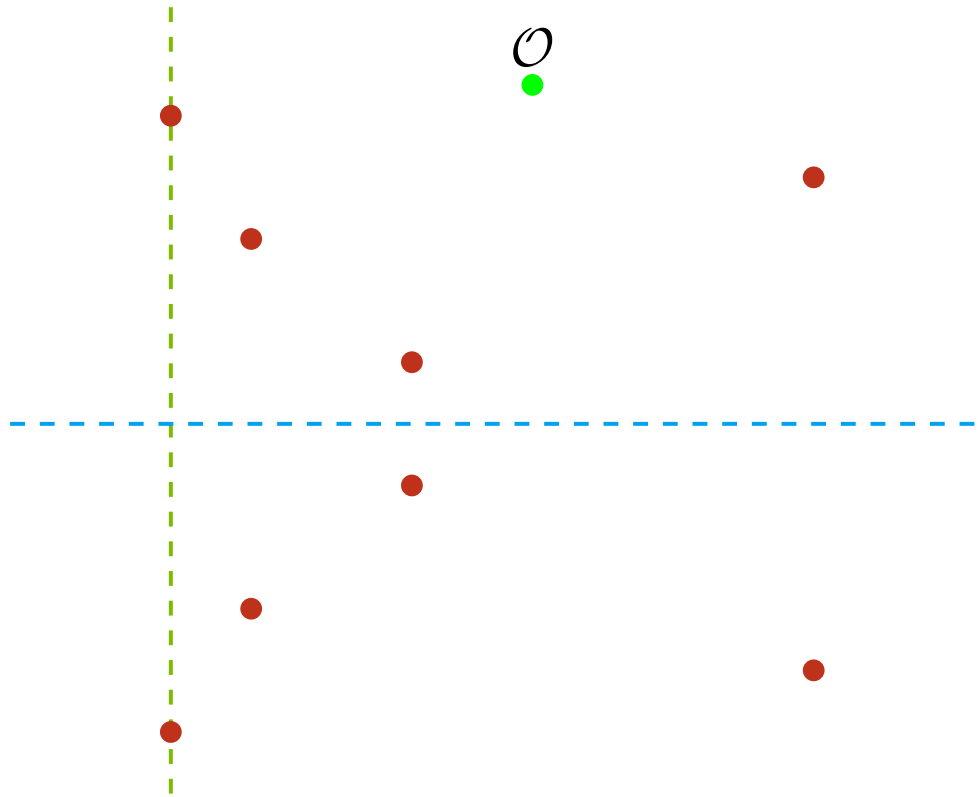

Figure: $E/\mathbb{F}_{11} : y^2 = x^3 + 5x + 3$

$L_1 : X = 0$

$\mathcal{A}_3 = \left(X_3^{(3)}, Y_3^{(3)}, Z_3^{(3)}\right)$, where

$$
\begin{aligned}
X_3^{(3)} &= (X_1Y_2 + X_2Y_1)(X_1Y_2 - X_2Y_1) \\
&\quad + a_4X_1X_2(X_1Z_2 - X_2Z_1) + 3a_6(X_1Z_2 + X_2Z_1)(X_1Z_2 - X_2Z_1) \\
&\quad - a_4^2(X_1Z_2 - X_2Z_1)Z_1Z_2, \\
Y_3^{(3)} &= (X_1Y_2 - X_2Y_1)Y_1Y_2 - 3a_4X_1X_2(Y_1Z_2 - Y_2Z_1) \\
&\quad + a_4(X_1Y_2 + X_2Y_1)(X_1Z_2 - X_2Z_1) + 3a_6(X_1Y_2 - X_2Y_1)Z_1Z_2 \\
&\quad - 3a_6(X_1Z_2 + X_2Z_1)(Y_1Z_2 - Y_2Z_1) + a_4^2(Y_1Z_2 - Y_2Z_1)Z_1Z_2, \\
Z_3^{(3)} &= -(X_1Y_2 + X_2Y_1)(Y_1Z_2 - Y_2Z_1) - (X_1Z_2 - X_2Z_1)Y_1Y_2 \\
&\quad - a_4(X_1Z_2 + X_2Z_1)(X_1Z_2 - X_2Z_1) - 3a_6(X_1Z_2 - X_2Z_1)Z_1Z_2.
\end{aligned}
$$

$L_2 : Y = 0$

$\mathcal{A}_2 = \left(X_3^{(2)}, Y_3^{(2)}, Z_3^{(2)}\right)$, where

$$
\begin{aligned}
X_3^{(2)} &= Y_1 Y_2 (X_1 Y_2 + X_2 Y_1) - a_4 (X_1 X_2 (Y_1 Z_2 + Y_2 Z_1) \\
&\quad - a_4 (X_1 Y_2 + X_2 Y_1)(X_1 Z_2 + X_2 Z_1) - 3a_6 (X_1 Y_2 + X_2 Y_1) Z_1 Z_2 \\
&\quad - 3a_6 (X_1 Z_2 + X_2 Z_1)(Y_1 Z_2 + Y_2 Z_1) + a_4^2 (Y_1 Z_2 + Y_2 Z_1) Z_1 Z_2, \\
Y_3^{(2)} &= Y_1^2 Y_2^2 + 3a_4 X_1^2 X_2^2 + 9a_6 X_1 X_2 (X_1 Z_2 + X_2 Z_1) \\
&\quad - a_4^2 X_1 Z_2 (X_1 Z_2 + 2X_2 Z_1) + a_4^2 (X_1 Z_2 + X_2 Z_1)(X_1 Z_2 - X_2 Z_1) \\
&\quad - 3a_4 a_6 X_1 Z_2 Z_2^2 - 3a_4 a_6 X_2 Z_1^2 Z_2 - (a_4^3 + 9a_6^2) Z_1^2 Z_2^2. \\
Z_3^{(2)} &= 3X_1 X_2 (X_1 Y_2 + X_2 Y_1) + Y_1 Y_2 (Y_1 Z_2 + Y_2 Z_1) \\
&\quad + a_4 (X_1 Y_2 + X_2 Y_1) Z_1 Z_2 + a_4 (X_1 Z_2 + X_2 Z_1)(Y_1 Z_2 + Y_2 Z_1) \\
&\quad + 3a_6 (Y_1 Z_2 + Y_2 Z_1) Z_1 Z_2.
\end{aligned}
$$

$L_3 : Z = 0$

$$\mathcal{A}_1 = \left(X_3^{(1)}, Y_3^{(1)}, Z_3^{(1)}\right), \text{ where}$$

$$\begin{aligned}
X_3^{(1)} &= (X_1Y_2 - X_2Y_1)(Y_1Z_2 + Y_2Z_1) + (X_1Z_2 - X_2Z_1)Y_1Y_2 \\
&\quad - a_4(X_1Z_2 + X_2Z_1)(X_1Z_2 - X_2Z_1) - 3a_6(X_1Z_2 - X_2Z_1)Z_1Z_2, \\
Y_3^{(1)} &= -3X_1X_2(X_1Y_2 - X_2Y_1) - Y_1Y_2(Y_1Z_2 - Y_2Z_1) \\
&\quad - a_4(X_1Y_2 - X_2Y_1)Z_1Z_2 + a_4(X_1Z_2 + X_2Z_1)(Y_1Z_2 - Y_2Z_1) \\
&\quad + 3a_6(Y_1Z_2 - Y_2Z_1)Z_1Z_2, \\
Z_3^{(1)} &= 3X_1X_2(X_1Z_2 - X_2Z_1) - (Y_1Z_2 + Y_2Z_1)(Y_1Z_2 - Y_2Z_1) \\
&\quad + a_4(X_1Z_2 - X_2Z_1)Z_1Z_2.
\end{aligned}$$

An explicit correspondence

Find an explicit addition law for any $(a : b : c) \in \mathbb{P}^2$, by

$$\begin{aligned}(a : b : c) &\leftrightarrow L : aX + bY + cZ = 0 \\ &\leftrightarrow a\mathcal{A}_3 + b\mathcal{A}_2 + c\mathcal{A}_1,\end{aligned}$$

which is complete if and only if $L \cap E = \emptyset$

Which is best?

Intuitive arguments

- $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ contain (mostly) distinct monomials
  - $\Rightarrow$ No cancellation occurs
  - $\Rightarrow$ Should choose a, b and/or c to be 0
  - $\Rightarrow$ The choice $a = c = 0$, $b = 1$ is complete on *odd* curves
- End up with $\mathcal{A}_2$ corresponding to $L : Y = 0$
- Has been considered, but inefficient!

Appearance in [AKR12]

[AKR12, Remark 4.4]

[...] the sum $(X_1 : Y_1 : Z_1) + (X_2 : Y_2 : Z_2)$ is given by the addition law $(X_3^{(2)}, Y_3^{(2)}, Z_3^{(2)})$ of Bosma and Lenstra:

$$\begin{aligned}
X_3^{(2)} &= (X_1Y_2 + Y_1X_2)(Y_1Y_2 - 6bZ_1Z_2) - X_1Z_2(aX_1Y_2 + 3bY_1Z_2) \\
&\quad - Z_1X_2(aY_1X_2 + 3bZ_1Y_2) - a(Y_1Z_2 + Z_1Y_2)(2X_1X_2 - aZ_1Z_2), \\
Y_3^{(2)} &= Y_1^2Y_2^2 + aX_1X_2(3X_1X_2 - 2aZ_1Z_2) - a^2(X_1Z_2 + Z_1X_2)^2 \\
&\quad + 3b(X_1Z_2 + Z_1X_2)(3X_1X_2 - aZ_1Z_2) - (a^3 + 9b^2)Z_1^2Z_2^2, \\
Z_3^{(2)} &= Y_1Y_2(Y_1Z_2 + Z_1Y_2) + (3X_1X_2 + 2aZ_1Z_2)(X_1Y_2 + Y_1X_2) \\
&\quad + (aX_1 + 3bZ_1)Y_1Z_2^2 + Z_1^2(aX_2 + 3bZ_2)Y_2,
\end{aligned}$$

specialized to $(a_1, a_2, a_3, a_4, a_6) = (0, 0, 0, a, b)$. [...]

Finding the structure

Rewrite $\mathcal{A}_2$ for more serious optimization:

$$
\begin{aligned}
X_3 &= (X_1Y_2 + X_2Y_1)(Y_1Y_2 - a(X_1Z_2 + X_2Z_1) - 3bZ_1Z_2) \\
&\quad - (Y_1Z_2 + Y_2Z_1)(aX_1X_2 + 3b(X_1Z_2 + X_2Z_1) - a^2Z_1Z_2), \\
Y_3 &= (Y_1Y_2 + a(X_1Z_2 + X_2Z_1) + 3bZ_1Z_2)(Y_1Y_2 - a(X_1Z_2 + X_2Z_1) - 3bZ_1Z_2) \\
&\quad + (3X_1X_2 + aZ_1Z_2)(aX_1X_2 + 3b(X_1Z_2 + X_2Z_1) - a^2Z_1Z_2), \\
Z_3 &= (Y_1Z_2 + Y_2Z_1)(Y_1Y_2 + a(X_1Z_2 + X_2Z_1) + 3bZ_1Z_2) \\
&\quad + (X_1Y_2 + X_2Y_1)(3X_1X_2 + aZ_1Z_2).
\end{aligned}
$$

Can we do better?

Can we do better? Maybe!

Very little is proven about optimality

- Is this the optimal way to compute the $Y = 0$ addition law?
- Are there more optimal complete addition laws?
  - For prime order curves?
  - For other curves?
- Are there more optimal incomplete addition laws?
  - Faster than currently used homogeneous addition law
- Different coordinate systems?
  - Jacobian?
  - Others?
- Higher bidegrees (μ, ν) for $\mu, \nu \geq 3$

Thanks

Thanks for your attention

References I

[Acc99a] Accredited Standards Committee X9. *American National Standard X9.62-1999, Public key cryptography for the financial services industry: the elliptic curve digital signature algorithm (ECDSA).* Draft at `http://grouper.ieee.org/groups/1363/Research/Other.html`. 1999.

[Acc99b] Accredited Standards Committee X9. *American National Standard X9.63-2001, Public key cryptography for the financial services industry: key agreement and key transport using elliptic curve cryptography.* Draft at `http://grouper.ieee.org/groups/1363/Research/Other.html`. 1999.

[AKR12] C. Arene, D. Kohel and C. Ritzenthaler. "Complete addition laws on abelian varieties". In: *LMS Journal of Computation and Mathematics* 15 (2012), pp. 308–316.

References II

[AR14] Hamad Alrimeih and Daler Rakhmatov. "Fast and Flexible Hardware Support for ECC Over Multiple Standard Prime Fields". In: *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 22.12 (Dec. 2014), pp. 2661–2674. ISSN: 1063-8210.

[Bal+12] Brian Baldwin, Raveen R. Goundar, Mark Hamilton and William P. Marnane. "Co-Z ECC scalar multiplications for hardware, software and hardware–software co-design on embedded systems". In: *Journal of Cryptographic Engineering* 2.4 (2012), pp. 221–240. ISSN: 2190-8516. DOI: 10.1007/s13389-012-0042-2. URL: %5Chref%7B%7Bhttp://dx.doi.org/10.1007/s13389-012-0042-2%7D%7D%7Bhttp://dx.doi.org/10.1007/s13389-012-0042-2%7D.

References III

[BJ02] E. Brier and M. Joye. "Weierstraß Elliptic Curves and Side-Channel Attacks". In: *Public Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002, Paris, France, February 12-14, 2002, Proceedings*. Ed. by D. Naccache and P. Paillier. Vol. 2274. Lecture Notes in Computer Science. Springer, 2002, pp. 335–345. ISBN: 3-540-43168-3. DOI: 10.1007/3-540-45664-3_24. URL: http://dx.doi.org/10.1007/3-540-45664-3_24.

[BL09] D. J. Bernstein and T. Lange. *Complete addition laws for elliptic curves*. Talk at Algebra and Number Theory Seminar (Universidad Autonomo de Madrid). Slides at http://cr.yp.to/talks/2009.04.17/slides.pdf. 2009.

[BL95] W. Bosma and H. W. Lenstra. "Complete systems of two addition laws for elliptic curves". In: *Journal of Number theory* 53.2 (1995), pp. 229–240.

References IV

[Bos+15] Joppe W. Bos, Craig Costello, Patrick Longa and Michael Naehrig. "Selecting Elliptic Curves for Cryptography: An Efficiency and Security Analysis". In: *J. Cryptographic Engineering* (2015). http://dx.doi.org/10.1007/s13389-015-0097-y. DOI: 10.1007/s13389-015-0097-y.

[Cer10] Certicom Research. *SEC 2: Recommended Elliptic Curve Domain Parameters, Version 2.0*. http://www.secg.org/sec2-v2.pdf. 2010.

[Cer15] Certivox UK, Ltd. *CertiVox Standard Curves*. http://docs.certivox.com/docs/miracl/certivox-standard-curves. Date accessed: September 9, 2015.

[ECC05] ECC Brainpool. *ECC Brainpool Standard Curves and Curve Generation*. http://www.ecc-brainpool.org/download/Domain-parameters.pdf. 2005.

References V

[GP08] Tim Güneysu and Christof Paar. "Ultra High Performance ECC over NIST Primes on Commercial FPGAs". In: *Cryptographic Hardware and Embedded Systems – CHES 2008*. Vol. 5154. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, pp. 62–78. ISBN: 978-3-540-85052-6.

[Gui10] Nicolas Guillermin. "A High Speed Coprocessor for Elliptic Curve Scalar Multiplications over $\mathbb{F}_p$". In: *Cryptographic Hardware and Embedded Systems, CHES 2010: 12th International Workshop, Santa Barbara, USA, August 17-20, 2010. Proceedings*. Vol. 6225. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 48–64. ISBN: 978-3-642-15031-9.

[LK15] K. C. C. Loi and S. B. Ko. "Scalable Elliptic Curve Cryptosystem FPGA Processor for NIST Prime Curves". In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 23.11 (Nov. 2015), pp. 2753–2756. ISSN: 1063-8210. DOI: 10.1109/TVLSI.2014.2375640.

References VI

[LR85] H. Lange and W. Ruppert. "Complete systems of addition laws on abelian varieties". In: *Inventiones mathematicae* 79.3 (1985), pp. 603–610.

[Ma+14] Yuan Ma, Zongbin Liu, Wuqiong Pan and Jiwu" Jing. "A High-Speed Elliptic Curve Cryptographic Processor for Generic Curves over $\mathrm{GF(p)}$". In: *Selected Areas in Cryptography – SAC 2013: 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 421–437. ISBN: 978-3-662-43414-7. DOI: 10.1007/978-3-662-43414-7_21.

[MMM06] Ciaran McIvor, Máire McLoone and John V. McCanny. "Hardware Elliptic Curve Cryptographic Processor Over $\mathrm{GF(p)}$". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 53.9 (Sept. 2006), pp. 1946–1957. ISSN: 1549-8328.

[MRB16] Pedro Maat C. Massolino, Joost Renes and Lejla Batina. *Implementing Complete Formulas on Weierstrass Curves in Hardware*. Cryptology ePrint Archive, Report 2016/1133. http://eprint.iacr.org/2016/1133. 2016.

References VII

[Nat13] National Institute for Standards and Technology. *Federal Information Processing Standards Publication 186-4. Digital signature standard*. Tech. rep. NIST, 2013.

[PMG14] Christopher Pöpper, Oliver Mischke and Tim Güneysu. "MicroACP - A Fast and Secure Reconfigurable Asymmetric Crypto-Processor". In: *Reconfigurable Computing: Architectures, Tools, and Applications*. Vol. 8405. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 240–247. ISBN: 978-3-319-05959-4.

[RDM15] Debapriya Basu Roy, Poulami Das and Debdeep Mukhopadhyay. "ECC on Your Fingertips: A Single Instruction Approach for Lightweight ECC Design in GF (p)". In: *Selected Areas in Cryptography - 22nd International Conference, Sackville, NB, Canada, August 12-14, 2015, Revised Selected Papers*. Springer International Publishing, 2015. ISBN: 978-3-319-13051-4.

References VIII

[SG14] Pascal Sasdrich and Tim Güneysu. "Efficient Elliptic-Curve Cryptography Using Curve25519 on Reconfigurable Devices". In: *Reconfigurable Computing: Architectures, Tools, and Applications*. Vol. 8405. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 25–36. ISBN: 978-3-319-05959-4.

[VGM11] Michal Varchola, Tim Güneysu and Oliver Mischke. "MicroECC: A Lightweight Reconfigurable Elliptic Curve Crypto-processor". In: *Reconfigurable Computing and FPGAs (ReConFig), 2011 International Conference on*. Nov. 2011, pp. 204–210.

[Vli+10] Jo Vliegen, Nele Mentens, Jan Genoe, An Braeken, Serge Kubera, Abdellah Touhafi and Ingrid Verbauwhede. "A compact FPGA-based architecture for elliptic curve cryptography over prime fields". In: *Application-specific Systems Architectures and Processors (ASAP), 2010 21st IEEE International Conference on*. July 2010, pp. 313–316.