Implementation of elliptic curve cryptography in constrained environments

Anthony Van Herrewege
Prof. Dr. Ir. I. Verbauwhede & Prof. Dr. Ir. B. Preneel

18 Februari 2009

Outline

Outline

Implement a compact hardware implementation of elliptic curve pairings.

Implement a compact hardware implementation of elliptic curve pairings.

- Program in GEZEL
- Optimize in VHDL
- Synthetize to FPGA/ASIC

Outline

Overview

1. What?
2. Why?
3. How?

What?

- Public key cryptography

What?

- Public key cryptography
- Identity-based cryptography

What?

- Public key cryptography
- Identity-based cryptography
- Calculations over elliptic curves

Why?

- Identity-based cryptography
  - No public key lookup required:
    eg. $P =$ National identification number

Why?

- Identity-based cryptography
  - No public key lookup required:
    eg. $P =$ National identification number
  - Date-stamped encryption possible:
    eg. $P =$ Nin $+$ "20091223"

Why?

- Identity-based cryptography
  - No public key lookup required:
    eg. $P =$ National identification number
  - Date-stamped encryption possible:
    eg. $P =$ Nin + "20091223"
  - Other positive aspects:
    Non-interactive key establishment
    Single round tripartite key establishment
    Ideal for eg. sensor networks

Why?

- Identity-based cryptography
  - No public key lookup required:
    eg. $P =$ National identification number
  - Date-stamped encryption possible:
    eg. $P =$ Nin + "20091223"
  - Other positive aspects:
    Non-interactive key establishment
    Single round tripartite key establishment
    Ideal for eg. sensor networks
  - Drawbacks as well:
    no key revocation, still a central authority, ...

Why?

- Identity-based cryptography
  - No public key lookup required:
    eg. $P =$ National identification number
  - Date-stamped encryption possible:
    eg. $P =$ Nin + "20091223"
  - Other positive aspects:
    Non-interactive key establishment
    Single round tripartite key establishment
    Ideal for eg. sensor networks
  - Drawbacks as well:
    no key revocation, still a central authority, ...
- Key strength comparison [bits]:

RSA	3072
ECC	256

How?

Elliptic curve pairing e:

$$e : G_1 \times G_1 \rightarrow G_2$$

Mapping needs to be:

- Bilinear: $e(P_1 + P_2, P_3) = e(P_1, P_3) \cdot e(P_2, P_3)$
- Non-degenerate: $e(P, P) \neq 1$
- Efficiently computable

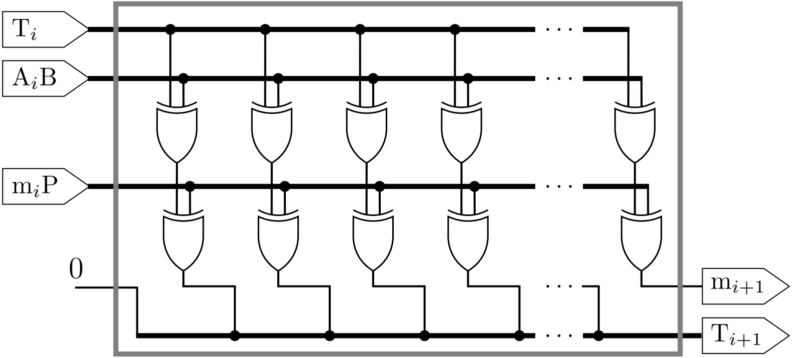
Several available pairings:

Weil, Tate, ate, eta, ...

Outline

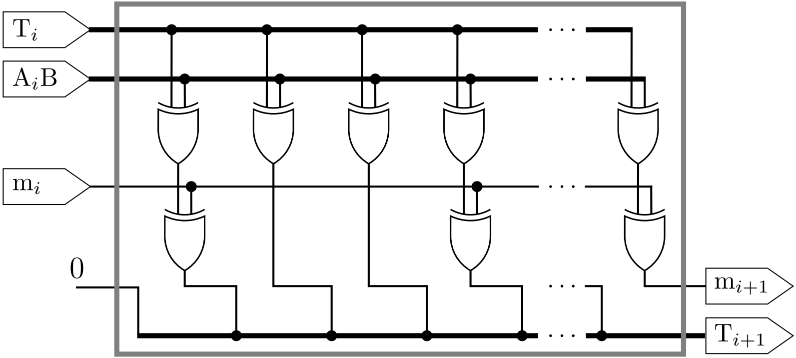
MALU

Modulo Arithmetic Logical Unit [general]:



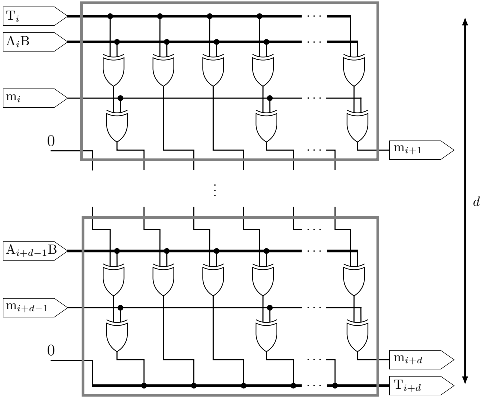
MALU

Modulo Arithmetic Logical Unit [optimized]:



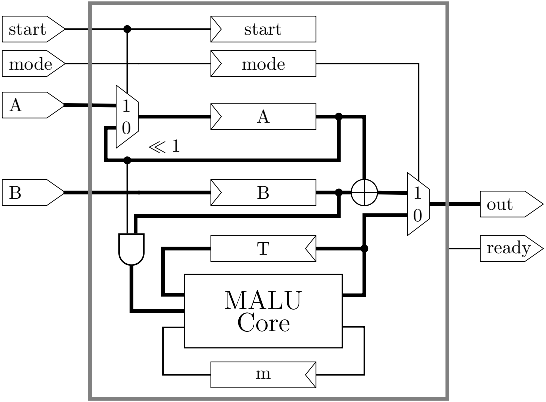
MALU

Modulo Arithmetic Logical Unit [optimized; d-bits wide]:



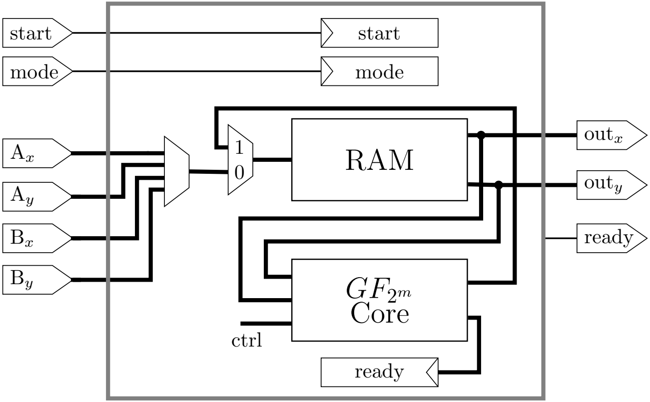
Wrappers - GF_{2^m}

GF_{2^m} Multiplication/Addition:



Wrappers - ECC

ECC Point Addition/Doubling:



State of the art

Some currently available implementations:

Name	Platform	Field	Speed
TinyTate	ATMega128L [7.4Mhz]	$\mathbb{F}_{2^{256}}$	30.2s
TinyPBC	ATMega128L [7.4Mhz]	$\mathbb{F}_{2^{256}}$	5.45s
Hankerson	P4 [2.8Ghz]	$\mathbb{F}_{2^{1223}}$	0.07s
Hankerson	P4 [2.8Ghz] (SSE)	$\mathbb{F}_{2^{1223}}$	0.03s

Outline

Progress so far

- MALU

Progress so far

- MALU
- GF_{2^m} functions

Progress so far

- MALU
- GF_{2^m} functions
- ECC functions

Progress so far

- MALU
- GF_{2^m} functions
- ECC functions
- Pairing functions (partial)

To do

- Complete pairing functions

To do

- Complete pairing functions
- Bugfixing

To do

- Complete pairing functions
- Bugfixing
- Optimization (VHDL)

To do

- Complete pairing functions
- Bugfixing
- Optimization (VHDL)
- Write thesis text

The end

Questions?