Arithmetic operators on $GF(2^m)$ for cryptographic applications: performance – power consumption – security tradeoffs

Danuta Pamuła

17th December 2012
1. **Introduction**

- Arithmetic operators on $GF(2^m)$ - application, requirements
- Arithmetics in $GF(2^m)$ and elliptic curve cryptography
- Formulated thesis
Arithmetic operators on $GF(2^m)$ - applications

- Cryptography:
  - symmetric: AES, ...
  - asymmetric: RSA, ...
  - Elliptic Curve Cryptography (ECC)

- error correcting codes
- computational biology (e.g. modelisation of genetic network)
- computational and algorithmic aspects of commutative algebra
- digital signal processing

...
Arithmetics in $GF(2^m)$ and ECC

- finite-field arithmetics ($GF(p)$, $GF(2^m)$)
- $2P$, $P+Q$
- $[k]P$
- encryption, digital signature

Danuta Pamuła
Cryptosystem - requirements

- speed/operaion time
- size/area
- power consumption

EFFICIENCY/PERFORMANCE
SECURITY

- mathematical attacks
- physical attacks
- perturbation
- observation

Danuta Pamuła

Arithmetic operators on $GF(2^m)$
Security of ECC systems

e.g. ElGamal encryption

Double-and-Add

for \( i = 0 \) to \( m-1 \)
if \( k_i = 1 \) then \( Q = Q + P \);  // (ADD)
\( P = 2P \);  // (DBL)

\[
R(x_R, y_R) = P(x_p, y_p) + Q(x_Q, y_Q)
\]
\[
X_R = \left( \frac{y_p + y_Q}{x_p + x_Q} \right)^2 + \frac{y_p + y_Q}{x_p + x_Q} + x_p + x_Q + a;
\]
\[
Y_R = \left( \frac{y_p + y_Q}{x_p + x_Q} \right) (x_R + x_p) + x_R + y_p;
\]
It is possible to create efficient and secure against some side-channel power analysis attacks $GF(2^m)$ arithmetic operators dedicated to reconfigurable hardware.
2. Arithmetic in $GF(2^m)$ - efficient and secure hardware solutions

- Basics
- Addition
- Multiplication
- Proposed solutions
Arithmetics in $GF(2^m)$

PARAMETERS

- basis (element representation)
- field size $m$
- irreducible polynomial $f(x)$ (field generator)

standard
- normal, GNB, ONB,
- dual

NIST, SECG
- cryptographic standards
  (FIPS 186-3, SEC 1, SEC 2)

GNB, ONB - Gaussian/Optimal Normal Basis, NIST - National Institute of Standards and Technology, SECG - Standards for Efficient Cryptography Group
Addition in $GF(2^m)$

Addition = XOR of binary polynomials

$c = a \ XOR \ b$

**Propositions** *(data in processor are passed in words (16, 32-bit):*

[1/2] Add every two incoming words of $a, b$, accumulate partial results in register $c$ (1) or in BlockRAM (2);

[3] Wait for all words of $a, b$, add $a$ and $b$;

<table>
<thead>
<tr>
<th>field size $m$</th>
<th>(1)(Virtex-6)</th>
<th>(2)(Virtex-6)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[LUT] [MHz]</td>
<td>[LUT] [MHz]</td>
</tr>
<tr>
<td>163</td>
<td>21  771</td>
<td>26  562</td>
</tr>
<tr>
<td>233</td>
<td>21  771</td>
<td>26  562</td>
</tr>
<tr>
<td>283</td>
<td>22  767</td>
<td>28  560</td>
</tr>
<tr>
<td>409</td>
<td>22  767</td>
<td>28  560</td>
</tr>
<tr>
<td>571</td>
<td>24  578</td>
<td>31  558</td>
</tr>
</tbody>
</table>
Multiplication in $GF(2^m)$

$c(x) = a(x)b(x) \mod f(x)$

two-step algorithms

interleaved algorithms

divide and conquer

reduction

matrix-vector

classic

matrix

R

Montgomery

Mastrovito
Multiplication - Mastrovito matrix approach

Idea:

\[ c = M b, \]

where \( M \) is a \( m \times m \) Mastrovito matrix

Problems:

1. **Size** of matrix \( M \) (\( m = 163, 233, 283, 409, 571 \))
2. **Construction** of matrix \( M \) (iterative algorithm, combination of matrices \( A \) and \( R \))
3. **Storing** matrix \( M \)
4. **Multiplication** of matrix \( M \) by vector \( b \)
### Arithmetic in $GF(2^m)$

#### Basics

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>Mc</td>
</tr>
<tr>
<td>1</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
</tr>
<tr>
<td>2</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
</tr>
<tr>
<td>3</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M1</td>
<td>Mc</td>
</tr>
<tr>
<td>4</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>5</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>6</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>7</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>8</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>9</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>10</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>11</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>12</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>13</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>MC</td>
</tr>
<tr>
<td>14</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>Mc</td>
</tr>
</tbody>
</table>

#### Summary - results, comments, future prospects

- $c_{1} = M_{0}[M(0,0),b(0)] + ... + M_{1}[M(0,13),b(0)]$
- $c_{2} = M_{0}[M(1,0),b(1)] + ... + M_{1}[M(1,13),b(1)]$
- $c_{5} = M_{2}[M(5,0),b(5)] + ... + M_{4}[M(5,13),b(5)]$
- $c_{14} = M_{R}[M(14,0),b(14)] + ... + M_{R}[M(14,14),b(14)]$

Danuta Pamuła

Arithmetic operators on $GF(2^m)$
1. **Partition of** $M$ **into** submatrices $16 \times 16$ bit
2. **Construction** of submatrices “on-the-fly” during multiplication, determination of submatrices with similar structures
3. Specialised **submultipliers** for each submatrix type - submultiplier constructs required submatrix during multiplication
4. The schedule of multiplication $M(i,j)b(i)$ is controlled by **Finite State Machine (FSM)**
Security of the operator - power (activity) analysis

Mastrovito unprotected - activity traces

Mastrovito unprotected - current measurements

Danuta Pamuła
Arithmetic operators on $GF(2^m)$
Increasing security against power cryptanalysis

- **uniformisation**
- **randomisation**
- **initialisation/reinitialisation**
- **optimization**
- **dummy operations**
- **FSM**
- **sub-multipliers**
- **BlockRam**

Note/Constraint: mainly algorithmic modifications, strictly hardware modifications were not considered (portability of the solution)
Optimization

most optimizations/decomposition left to synthesis tool

Proposition: (optimization/decomposition (if possible) “by hand”)

- removal of auxiliary/unnecessary registers;
- partitioning of very large registers and complex, sequential operations into smaller/easier(simpler) ones;
- merging sequential operations

BlockRam

only LUT blocks were used to implement solutions

Proposition: units were partially implemented in BlockRams - according to some sources it diminishes power consumption;
Introduction
Arithmetic in $GF(2^m)$
Summary - results, comments, future prospects

Basics
Addition
Multiplication

1399 LUT
571MHz

$e_1 = d[2m-2,...,m]$

$e_1f = e_1 \times f$

$d_1 = d \oplus e_1f$

$e_2 = d_1[74+(m-1),...,m]$

$e_2f = e_2 \times f$

$c = d_1 \oplus e_2f$

idle

350 LUT
810 MHz

$d[463,...,0]$
$c[232,...,0]$

Danuta Pamuła
Arithmetic operators on $GF(2^m)$
FSM

one FSM controlling all submultipliers, many states (necessity of re-utilisation of submultipliers)

Proposition:

- **uniformisation**: same number of states, unification of number of registers/bit switching in each state, changed order of submultiplications;

- **randomisation**: each instance/type of submultiplier is controlled by different FSM (additional FSMs), each FSM is started at different moment of multiplication process;

- less states, more instances of submultipliers used, more activity in one state (no submultiplier is idle in any state);

- avoiding idle states between consecutive multiplications;
### Introduction

Arithmetic in $GF(2^m)$

### Summary - results, comments, future prospects

### Basics

#### Addition

<table>
<thead>
<tr>
<th>$M$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
</tr>
<tr>
<td>1</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
</tr>
<tr>
<td>2</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
</tr>
<tr>
<td>3</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
<td>M1</td>
</tr>
<tr>
<td>4</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
</tr>
<tr>
<td>5</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
</tr>
<tr>
<td>6</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
<td>M4</td>
</tr>
<tr>
<td>7</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
</tr>
<tr>
<td>8</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
</tr>
<tr>
<td>9</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
</tr>
<tr>
<td>10</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
</tr>
<tr>
<td>11</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
</tr>
<tr>
<td>12</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
</tr>
<tr>
<td>13</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M0</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
<td>M2</td>
</tr>
<tr>
<td>14</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
<td>M_R</td>
</tr>
</tbody>
</table>

#### Multiplication

$c_1 = M_0[M(0,0),b(0)] + ... + M_1[M(0,13),b(0)];$
$c_2 = M_0[M(1,0),b(1)] + ... + M_1[M(1,13),b(1)];$
$c_3 = M_2[M(5,0),b(5)] + ... + M_4[M(5,13),b(5)];$
$c_{14} = M_R[M(14,0),b(14)] + ... + M_R[M(14,14),b(14)];$

**Danuta Pamuła**

Arithmetic operators on $GF(2^m)$
Submultipliers

One instance for each type of submatrix/submultiplier

Proposition:
- using more than one instance of the same submultiplier;
- note: submultipliers were optimised during efficiency analysis (by hand), these are combinational circuits;

Dummy operation

In some states some submultipliers are idle, some registers are unused

Proposition: dummy operations on unused registers;

Initialisation/reinitialisation

Registers are resetted/reloaded at the beginning of multiplication

Proposition:
- resetting/reloading just before use;
- filling with random values (not constant), instead of zeroes;
Introduction
Arithmetic in $GF(2^m)$
Summary - results, comments, future prospects

Basics
Addition
Multiplication

Danuta Pamuła
Arithmetic operators on $GF(2^m)$
## Introduction

Arithmetic in \( GF(2^m) \)

## Summary - results, comments, future prospects

### Basics

#### Addition

**Algorithm**

- **Mastrovito**
  - Area: 3760 LUTs
  - Frequency: 297 MHz
  - Clock cycles: 75
  - AT: 0.95

- **Mastrovito v0**
  - Area: 3889 LUTs (×1.03)
  - Frequency: 225 MHz (×0.75)
  - Clock cycles: 48 (×0.64)
  - AT: 0.83

- **Mastrovito v1**
  - Area: 3463 LUTs (×0.92)
  - Frequency: 414 MHz (×1.39)
  - Clock cycles: 75 (×1.00)
  - AT: 0.63

- **Mastrovito v2**
  - Area: 3700 LUTs (×0.98)
  - Frequency: 306 MHz (×1.03)
  - Clock cycles: avg.116 (×1.55)
  - AT: 1.35

- **Mastrovito v3**
  - Area: 3903 LUTs (×1.04)
  - Frequency: 319 MHz (×1.07)
  - Clock cycles: avg.80 (×1.07)
  - AT: 0.97

\( \alpha \): secured = \( \alpha \) × original

\( AT = \text{area} \times \text{execution_time} \)
5. **Summary - results, comments, future prospects**
Summarising, as a result of conducted researches the following original results were obtained:

- **efficient in terms of speed and area** $GF(2^m)$ hardware arithmetic operators dedicated to ECC applications were proposed:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Area [LUT]</th>
<th>$f$ [MHz]</th>
<th>clock cycles</th>
<th>$AT$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classic 1</td>
<td>3638</td>
<td>302</td>
<td>264</td>
<td>3.18</td>
</tr>
<tr>
<td>Classic 2</td>
<td>2862</td>
<td>302</td>
<td>238</td>
<td>2.25</td>
</tr>
<tr>
<td>Mastrovito</td>
<td>3760</td>
<td>297</td>
<td>75</td>
<td>0.95</td>
</tr>
<tr>
<td>Montgomery (full)</td>
<td>3197</td>
<td>338</td>
<td>270</td>
<td>2.55</td>
</tr>
</tbody>
</table>
• **successful protections** against some power analysis side channel attacks for $GF(2^m)$ hardware arithmetic operators were developed;

![Graphs showing number of transitions over cycles for different methods.](image)
the **tradeoff** between efficiency and security of $GF(2^m)$ hardware arithmetic operators was found.;

<table>
<thead>
<tr>
<th>Algorithm (Virtex 6)</th>
<th>area [LUT]</th>
<th>freq. [MHZ]</th>
<th># cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classical</td>
<td>2868</td>
<td>270</td>
<td>260</td>
</tr>
<tr>
<td>×α factor</td>
<td>×0.79</td>
<td>×0.89</td>
<td>×0.98</td>
</tr>
<tr>
<td>Montgomery</td>
<td>2099</td>
<td>323</td>
<td>264</td>
</tr>
<tr>
<td>×α factor</td>
<td>×0.96</td>
<td>×1.00</td>
<td>×0.98</td>
</tr>
<tr>
<td>Mastrovito v0</td>
<td>3889</td>
<td>225</td>
<td>48</td>
</tr>
<tr>
<td>×α factor</td>
<td>×1.03</td>
<td>×0.75</td>
<td>×0.64</td>
</tr>
<tr>
<td>Mastrovito v1</td>
<td>3463</td>
<td>414</td>
<td>75</td>
</tr>
<tr>
<td>×α factor</td>
<td>×0.92</td>
<td>×1.39</td>
<td>×1.00</td>
</tr>
<tr>
<td>Mastrovito v2</td>
<td>3700</td>
<td>306</td>
<td>avg.116</td>
</tr>
<tr>
<td>×α factor</td>
<td>×0.98</td>
<td>×1.03</td>
<td>×1.55</td>
</tr>
<tr>
<td>Mastrovito v3</td>
<td>3903</td>
<td>319</td>
<td>avg.80</td>
</tr>
<tr>
<td>×α factor</td>
<td>×1.04</td>
<td>×1.07</td>
<td>×1.07</td>
</tr>
</tbody>
</table>
## Arithmetic in $GF(2^m)$

### Summary - results, comments, future prospects

<table>
<thead>
<tr>
<th>Solution</th>
<th>$m$</th>
<th>FPGA</th>
<th>Area</th>
<th>max.$f$</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowe</td>
<td>256</td>
<td>Virtex II</td>
<td>5267 LUT</td>
<td>44.91 MHz</td>
<td>5.75 us</td>
</tr>
<tr>
<td>Grabbe</td>
<td>233</td>
<td>XC2V6000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FF1517-4</td>
<td>37296 LUT</td>
<td>77 MHz</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>11746 LUT</td>
<td>90.33 MHz</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>36857 LUT</td>
<td>62.85 MHz</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>45435 LUT</td>
<td>93.20 MHz</td>
<td>-</td>
</tr>
<tr>
<td>Rodriguez-Henrique</td>
<td>191</td>
<td>XCV2600E</td>
<td>8721 CLB</td>
<td></td>
<td>82.4 us</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$m$</th>
<th>FPGA</th>
<th>Area [LUT]</th>
<th>max.$f$ [MHz]</th>
<th>T [us]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classical mod</td>
<td>233</td>
<td>XC2V6000</td>
<td>4498</td>
<td>115</td>
<td>2.26</td>
</tr>
<tr>
<td>Montgomery</td>
<td></td>
<td></td>
<td>2099</td>
<td>129</td>
<td>2.04</td>
</tr>
<tr>
<td>Mastrovito v0</td>
<td></td>
<td></td>
<td>6387</td>
<td>183</td>
<td>0.26</td>
</tr>
<tr>
<td>Mastrovito v1</td>
<td></td>
<td></td>
<td>5154</td>
<td>107</td>
<td>0.7</td>
</tr>
<tr>
<td>Mastrovito v2</td>
<td></td>
<td></td>
<td>6364</td>
<td>113</td>
<td>1.02</td>
</tr>
<tr>
<td>Mastrovito v3</td>
<td></td>
<td></td>
<td>6387</td>
<td>100</td>
<td>0.8</td>
</tr>
</tbody>
</table>
Future prospects

- investigation of **inversion, division** in the field,
- investigation of **other** representations of elements of the field (**basis**) and its impact on the architecture,
- **integration with ECC processor** and further security evaluation
- **hardware countermeasures**: bus coding, clocks, special structures
- countermeasures against **other types of side-channel attacks**
- ...

Danuta Pamuła
Thank you for your attention
Dziękuję za uwagę
Merci pour votre attention
Thank you for your attention
Dziękuję za uwagę
Merci pour votre attention
Introduction

Arithmetic in $GF(2^m)$

Summary - results, comments, future prospects

---

**Danuta Pamuła**

Arithmetic operators on $GF(2^m)$