Hardware Implementation of the Elliptic Curve Method of Factoring

Master’s Thesis Presentation
Mohammed Khaleeluddin
Director: Dr. Kris Gaj
Contents

- Introduction
- ECM Algorithm
- Hardware Architecture
- Results
- Conclusions
In 1977

Ron Rivest, Adi Shamir & Leonard Adleman

developed the first public key cryptosystems, they called RSA
RSA

Public key \{e, N\} \quad \text{Private key \{d, P,Q\}}

Alice → Network → Bob

Encryption → Decryption

\{ e, N \} → \{ d, P, Q \}

N = P \cdot Q \quad \text{P, Q - large prime factors}

e \cdot d \equiv 1 \mod ((P-1)(Q-1))
Common Applications of RSA

Secure WWW, SSL

Browser ⟷ Network ⟷ WebServer

S/MIME, PGP

Alice ⟷ Bob
Recommended key sizes for RSA

Size of the RSA key = size of $N = P \cdot Q$

Old standard:
   Individual users
   512 bits (155 decimal digits)

New standard:
   Short-term use (up to 2010)
   1024 bits
   Long-term use
   2048 bits
Factoring RSA

RSA-200 (663-bits) factored by Bahr, Boehm, Frank and Kleinjung

When?
Dec 2003 – May 2005

Effort?
First stage:
   About 1 year on various machines, equivalent to 55 years on Opteron 2.2 GHz CPU

Second stage:
   3 months on a cluster of 80 2.2 GHz Opterons connected via a gigabit network
Number Field Sieve

Best Algorithm to Factor Large Numbers

Complexity: Sub-exponential time and memory

\[ N = \text{Number to factor}, \]
\[ k = \text{Number of bits of } N \]
ECM in Number Field Sieve (NFS)

1. Polynomial Selection
2. Relation Collection
3. Sieving 200-250 bit numbers
4. Trial Factoring (ECM)
5. Linear Algebra
6. Square Root

Hardware Implementation of the Elliptic Curve Method of Factoring
ECM Algorithm
What is ECM

Elliptic Curve Method of Factoring

Lenstra 1985 Phase 1
Brent, Montgomery 1986-87 Phase 2

$N$

$q$

< 50 bits

Factoring time depends mainly on the size of factor $q$
Elliptic Curve

- Not an ellipse
- Represented using cubic equations similar to those used for calculating the circumference of an ellipse

\[ Y^2 = X^3 + A \cdot X + B \]
### Elliptic Curve

\[ Y^2 = X^3 + X + 1 \mod 23 \]

**Points fulfilling the equation of the curve**

- **Addition**
  - \( P = (6, 19) \)
  - \( Q = (7, 12) \)
  - \( R = P + Q = (13, 7) \)
- **Doubling**
  - \( P = (3, 13) \)
  - \( 2P = P + P = (7, 11) \)

Diagram showing points on the curve and operations involving them.
Projective vs. Affine coordinates

- Affine coordinates
  
  $P_a = (x_P, y_P)$
  
  - Addition and doubling requires inversion.

- Projective coordinates
  
  $P_p = (x_P, y_P, z_P)$
  
  - Addition and doubling can be done without inversion.

- Projective coordinates for Montgomery form of curve
  
  $P_{pM} = (x_P : : z_P)$
  
  - Addition and doubling do not require $y$ coordinate.
Scalar Multiplication

\[ Q = k \cdot P = P + P + P + \ldots + P \]

- point
- number (scalar)
- point

\( k \)-times
**ECM Algorithm**

- **Inputs:**
  - \( N \) – number to be factored
  - \( P_0 \) – point of a curve \( E \) : initial point
  - \( B_1 \) – bound for Phase 1
  - \( B_2 \) – bound for Phase 2

- **Outputs:**
  - \( q \) – factor of \( N \), \( 1 < q < N \) or **FAIL**


**ECM Algorithm Phase 1**

1: \[ k \leftarrow \prod_{p_i} p_i^{e_i} \text{ such that } p_i \text{ - consecutive primes } \leq B_1 \]

\[ e_i \text{ - largest exponent such that } p_i^{e_i} \leq B_1 \]

2: \[ Q_0 \leftarrow kP_0 = (x_{Q_0} : z_{Q_0}) \]

**Precomputations**

3: \[ q \leftarrow \gcd(z_{Q_0}, N) \]

**Main computations**

4: if \( q > 1 \)
5: return \( q \) (factor of \( N \))
6: else
7: go to Phase 2
8: end if

**Post-computations**
Phase 1 Example

\[ N = 1\,740\,719 = 1279 \cdot 1361 \]

\[ E : y^2 = x^3 + 14x + 1 \pmod{1\,740\,719} \]

\[ P_0 = (5 : : 1) \]

\[ B_1 = 20 \]

\[ k = 2^4 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19 = 232\,792\,560 \]

\[ kP_0 = (707\,838 : : 1\,686\,279) \]

\[ \gcd(1\,686\,279, 1\,740\,719) = 1361 \]
ECM Algorithm Phase 2

09: \( d \leftarrow 1 \)
10: for each prime \( p \) in the range \( B_1 \) to \( B_2 \) do
11: \( (x_{pQ_0}, y_{pQ_0}, z_{pQ_0}) \leftarrow pQ_0 \)
12: \( d \leftarrow d \cdot z_{pQ_0} \pmod{N} \)
13: end for

14: \( q \leftarrow \gcd(d, N) \)
15: if \( q > 1 \) then
16: return \( q \)
17: else
18: return FAIL
19: end if
**ECM Algorithm Phase 2**

Based on Standard Continuation Algorithm

- Basic step of Phase 2 $pQ_0$
- $p$ Can be represented as $p = m \cdot D \pm j$
- We need to compute $mDQ_0$ and $jQ_0$
- $jQ_0$ can be pre-computed for all $j$ such that $1 \leq j \leq D/2$ and $\gcd(j, D) = 1$
Choice of D

B1 = 960  \hspace{1cm} B2 = 57,000

\[ D = 30 = 2 \cdot 3 \cdot 5 \]

\[ D = 210 = 2 \cdot 3 \cdot 5 \cdot 7 \]

- \[ D = 30 = 2 \cdot 3 \cdot 5 \]
  - j
  - mD
  - 4
  - 1 \hspace{0.5cm} 7 \hspace{0.5cm} 11 \hspace{0.5cm} 13

- \[ D = 210 = 2 \cdot 3 \cdot 5 \cdot 7 \]
  - j
  - mD
  - 1
  - 24
  - 103

- Prime_table
  - 6408 bits
  - 4361 of 1’s
  - 65% of 1’s

- Prime_table
  - 1 4361 of 1’s
  - 61% of 1’s

- Prime_table
  - 1 if \( p = m \cdot D - j \) is prime or \( p = m \cdot D + j \) is prime
  - 0 otherwise

- Prime_table
  - 271
  - 267

- Prime_table
  - 1900
  - 1869
  - 7476 bits
  - 4531 of 1’s
  - 61% of 1’s
Hierarchy of ECM Operations

- Top level
  - ECM
  - Scalar multiplication
  - Elliptic curve point operations

- Medium level
  - $k \cdot P$
  - Point doubling

- Low level
  - $P + Q$
  - Point addition
  - $2P$
  - Modular subtraction
  - $x \cdot y \mod p$
  - Modular multiplication
  - $x + y \mod p$
  - Modular addition

Host computer

Control unit

Functional units

Modular arithmetic (field operations)
Hardware Architecture
Hardware Implementation of the Elliptic Curve Method of Factoring

ECM architecture: Top-level view

- FPGA
- ECM Units
- Instruction memory
- Control Unit Phase1 & Phase2
- I/O
- Global memory
- RAM
- Host computer
Global Memory

Hardware Implementation of the Elliptic Curve Method of Factoring
Local Memory

### Phase 1

<table>
<thead>
<tr>
<th></th>
<th>31</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(R_0 = N)</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_1 = a_{24})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_2 = x_{P_0})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_3 = z_{P_0})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_4 = x_{P})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_5 = z_{P})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_6 = x_{Q})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_7 = z_{Q})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_8)</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_9)</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_{i0})</td>
<td></td>
</tr>
<tr>
<td>87</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Phase 2

<table>
<thead>
<tr>
<th></th>
<th>31</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(R_0 = N/d)</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_1 = a_{24})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_2 = x_{Q_0})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_3 = z_{Q_0})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_4 = x_{P})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_5 = z_{P})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_6 = x_{Q})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_7 = z_{Q})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_8)</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_9)</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(R_{i0})</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(2Q_0)</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(jQ_0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[Q = DQ_0\]
\[R = mDQ_0\]
Scalar Multiplication – pseudo code

Input : $P_0 \in E (x_0 \neq 0), k = (k_{s-1}, k_{s-2}, \ldots, k_1, k_0)_2 \quad k_{s-1} = 1$
Output : $kP_0$

1: $Q \leftarrow P_0, P \leftarrow 2P_0$
2: for $i = s-2$ downto 0 do
3: \quad if $k_i = 1$ then
4: \quad \quad $Q \leftarrow P + Q, \quad P \leftarrow 2P$
5: \quad else
6: \quad \quad $Q \leftarrow 2Q, \quad P \leftarrow P + Q$
7: \quad end if;
8: end for
9: return $Q$
\[ Q \leftarrow P + Q, \quad P \leftarrow 2P \]

Input
\[ P = (x_P : z_P), \quad Q = (x_Q : z_Q), \quad P - Q = (x_{P-Q} : z_{P-Q}) \]

\[ P, Q, P - Q \in E, \quad a_{24} = \frac{a + 2}{4} \]

where \( a \) is a parameter of the curve \( E \)

Output
\[ P + Q = (x_{P+Q} : z_{P+Q}), \quad 2P = (x_{2P} : z_{2P}) \]

\[ x_{P+Q} = z_{P-Q} \left( (x_p - z_p)(x_Q + z_Q) + (x_p + z_p)(x_Q - z_Q) \right)^2 \]

\[ z_{P+Q} = x_{P-Q} \left( (x_p - z_p)(x_Q + z_Q) - (x_p + z_p)(x_Q - z_Q) \right)^2 \]

\[ 4x_p z_p = (x_p + z_p)^2 - (x_p - z_p)^2 \]

\[ x_{2P} = (x_p + z_p)^2(x_p - z_p)^2 \]

\[ z_{2P} = 4x_p z_p \left( (x_p - z_p)^2 + a_{24} \cdot (4x_p z_p) \right) \]

**Point addition**

- 6 multiplications when \( z_{P-Q} \neq 1 \)
- 5 multiplications when \( z_{P-Q} = 1 \)

**Point doubling**

- 5 multiplications
Hardware Implementation of the Elliptic Curve Method of Factoring

Computation Flow

\[ Q \leftarrow P + Q, \quad P \leftarrow 2P \]

<table>
<thead>
<tr>
<th>Adder/Subtractor</th>
<th>Multiplier 1</th>
<th>Multiplier 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>A/D: ( a_1 = x_p + z_p )</td>
<td>( m_1 = s_1^2 )</td>
<td>( m_2 = a_1^2 )</td>
</tr>
<tr>
<td>A/D: ( s_1 = x_p - z_p )</td>
<td>( m_2 = a_1^2 )</td>
<td>( m_2 = a_1^2 )</td>
</tr>
<tr>
<td>A/D: ( a_2 = x_Q + z_Q )</td>
<td>( m_1 = s_1^2 )</td>
<td>( m_2 = a_1^2 )</td>
</tr>
<tr>
<td>A/D: ( s_2 = x_Q - z_Q )</td>
<td>( m_2 = a_1^2 )</td>
<td>( m_2 = a_1^2 )</td>
</tr>
<tr>
<td>D: ( s_3 = m_2 - m_1 )</td>
<td>A: ( m_3 = s_1 \cdot a_2 )</td>
<td>A: ( m_4 = s_2 \cdot a_1 )</td>
</tr>
<tr>
<td>A: ( a_3 = m_3 + m_4 )</td>
<td>( m_4 = s_2 \cdot a_1 )</td>
<td>( m_4 = s_2 \cdot a_1 )</td>
</tr>
<tr>
<td>( s_4 = m_3 - m_4 )</td>
<td>D: ( x_{2P} = m_5 = m_1 \cdot m_2 )</td>
<td>D: ( m_6 = s_3 \cdot a_{24} )</td>
</tr>
<tr>
<td>D: ( a_4 = m_1 + m_6 )</td>
<td>A: ( x_{P+Q} = m_7 = a_3^2 )</td>
<td>A: ( m_8 = s_4^2 )</td>
</tr>
<tr>
<td>A: ( z_{P+Q} = m_9 = m_8 \cdot x_{P-Q} )</td>
<td>D: ( z_{2P} = m_{10} = s_3 \cdot a_4 )</td>
<td></td>
</tr>
</tbody>
</table>
Resource utilization in time

Time

Control Unit (8%)

MUL 2 (43%)

MUL 1 (43%)

ADD/SUB (6%)

Area

100%
Hardware Implementation of the Elliptic Curve Method of Factoring

Hierarchy of ECM Operations

- Top level: ECM
- Medium level: $k \cdot P$, $P+Q$, $2P$
- Low level: $x \cdot y \mod p$, $x+y \mod p$, $x-y \mod p$

- Host computer
- Control unit
- Functional units
- Scalar multiplication
- Point addition
- Point doubling
- Modular multiplication
- Modular addition
- Modular subtraction

Elliptic curve point operations

Modular arithmetic (field operations)
Montgomery Multiplication

Based on
McIvor, McLoone, et al.
Asilomar 2003:
full-length CSAs
word-length CPAs

Hardware Implementation of the Elliptic Curve Method of Factoring
Hardware Implementation of the Elliptic Curve Method of Factoring

Addition/Subtraction

Original design
ECM architecture: Top-level view
Control Unit

Phase 1

- Memory Initialization
- Scalar Multiplication
- Reading Out Results

Phase 2

- Memory Initialization
- Pre-Computations
- Main-Computations
- Reading Out Results
Control Unit

- Total 18 state machines with 197 states
  - 7 state machines with 46 states in Phase 1
  - 11 state machines with 151 states in Phase 2
- 4 Shift registers
- 14 Registers
- 10 Down counters
- 2 Up-down counters
- 25 Comparators

Original design
Example of one state machine – ASM chart
Results
# Families of Xilinx FPGA Devices

<table>
<thead>
<tr>
<th>Low-cost</th>
<th>High-performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spartan 3</td>
<td>Virtex II</td>
</tr>
<tr>
<td>(&lt; $130*)</td>
<td>(&lt; $2,700*)</td>
</tr>
<tr>
<td>Spartan 3E</td>
<td>Virtex 4</td>
</tr>
<tr>
<td>(&lt; $35*)</td>
<td>(&lt; $3,000*)</td>
</tr>
</tbody>
</table>

*approximate cost of the largest device per unit for a batch of 10,000 units
Number of ECM units per FPGA

<table>
<thead>
<tr>
<th>FPGA Type</th>
<th>Model</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spartan 3</td>
<td>XC3S5000-5</td>
<td>Low-cost</td>
</tr>
<tr>
<td></td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Virtex II</td>
<td>XC2V6000-6</td>
<td>High-performance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Spartan 3E</td>
<td>XC3S1600-5</td>
<td>Low-cost</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Virtex 4</td>
<td>XC4VLX200-11</td>
<td>High-performance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>27</td>
</tr>
</tbody>
</table>
Performance – ECM Operations per Second

- Spartan 3 (XC3S5000-5)
  - Low-cost
- Virtex II (XC2V6000-6)
  - High-performance
- Spartan 3E (XC3S1600-5)
  - Low-cost
- Virtex 4 (XC4VLX200-11)
  - High-performance

- 287
- 430
- 133
- 942

- x 1.5
- x 7.0

Hardware Implementation of the Elliptic Curve Method of Factoring
Performance to cost ratio

ECM Operations per second per $100

<table>
<thead>
<tr>
<th>Device</th>
<th>Operations per second per $100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spartan 3</td>
<td>Low-cost</td>
</tr>
<tr>
<td>XC3S5000-5</td>
<td>221</td>
</tr>
<tr>
<td>Virtex II</td>
<td>High-performance</td>
</tr>
<tr>
<td>XC2V6000-6</td>
<td>16</td>
</tr>
<tr>
<td>Spartan 3E</td>
<td>Low-cost</td>
</tr>
<tr>
<td>XC3S1600-5</td>
<td>380</td>
</tr>
<tr>
<td>Virtex 4</td>
<td>High-performance</td>
</tr>
<tr>
<td>XC4VLX200-11</td>
<td>31</td>
</tr>
</tbody>
</table>

Spartan 3: x 13.8
Virtex II: x 12.3
Previous Proof-of-Concept Design

Pelzl, Šimka, SHARCS Feb 2005
Kleinjung, Franke, FCCM Apr 2005
Priplata, Stahlke, IEE Proc. Oct 2005
Drutarovský, Fischer,
Paar
### Modifications Compared to Pelzl, Šimka, et al

#### Internal vs. External control

<table>
<thead>
<tr>
<th>Pelzl, Šimka</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td>host µC control ARM7</td>
<td>host control</td>
</tr>
<tr>
<td>FPGA</td>
<td>ECM units</td>
</tr>
<tr>
<td>ECM units</td>
<td></td>
</tr>
</tbody>
</table>

#### Memory management

<table>
<thead>
<tr>
<th>Pelzl, Šimka</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td>suboptimal use of memory space</td>
<td>bit tables &amp; consolidation of memory resources</td>
</tr>
</tbody>
</table>

#### Functional units

<table>
<thead>
<tr>
<th>Pelzl, Šimka</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td>MUL ADD/SUB</td>
<td>MUL1 MUL2 ADD/SUB</td>
</tr>
<tr>
<td>time</td>
<td></td>
</tr>
</tbody>
</table>

#### Montgomery multiplier

<table>
<thead>
<tr>
<th>Pelzl, Šimka</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td>Based on</td>
<td>Based on</td>
</tr>
<tr>
<td>Tenca, Koc</td>
<td>McIvor, McLoone et al</td>
</tr>
<tr>
<td>CHES 1999</td>
<td>Asilomar 2003</td>
</tr>
<tr>
<td>IEEE Trans. Comp. 2003</td>
<td>full-length CSAs</td>
</tr>
<tr>
<td>word-based CPA and/or CSA</td>
<td>word-length CPAs</td>
</tr>
</tbody>
</table>
Comparison with the Proof-of-Concept Design by Pelzl & Šimka

Equalizing Measures

- Use the same FPGA device (Xilinx Virtex 2000E-6)
- Pelzl and Šimka design assumed to be redesigned to include an internal controller. Execution times recalculated based on the limitations of the ECM unit only.
Modifications Compared to Pelzl, Šimka, et al

Internal vs. External Control
Pelzl, Šimka
New

Memory Management
Pelzl, Šimka
New
suboptimal use of memory space
bit tables & consolidation of memory resources

Functional Units
Pelzl, Šimka
New

Montgomery Multiplier
Pelzl, Šimka
New

- Based on
  - Tenca, Koc
  - CHES 1999
  - IEEE Trans. Comp. 2003
  - word-based CPA and/or CPA

- Based on
  - McIvor, McLoone et al
  - Asilomar 2003
  - full-length CSAs word-length CPAs
Comparison with the Proof-of-Concept Design by Pelzl & Šimka

**Timing**

### Phase 1
- **Pelzl/Šimka**
  - 293 ms
- **Factor of x 9.3**
  - 32 ms
- **New**

### Phase 2
- **Pelzl/Šimka**
  - 527 ms
- **Factor of x 7.4**
  - 72 ms
- **New**
  - 35 ms
- **D=30**
- **New**
  - **D=210**

**Major Contributors to the speed up:**

- Different design for the multiplier (x 5)
- Two multipliers working in parallel (x 1.9)
- Different D (x 1.9)
Comparison with the Proof-of-Concept Design by Pelzl & Šimka

**Resources**

<table>
<thead>
<tr>
<th></th>
<th>Memory (BRAMs)</th>
<th>Area (CLB Slices)</th>
<th>ECM Units / Virtex 2000E FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pelzl/Šimka</td>
<td>44 (27%)</td>
<td>6% (1.3%)</td>
<td>3 Limited by BRAMs</td>
</tr>
<tr>
<td>New</td>
<td>2 (1.3%)</td>
<td>15%</td>
<td>7 Limited by CLB Slices</td>
</tr>
</tbody>
</table>

- Factor of x 22
- Factor of x 2.5
- Factor of x 2.33
Modifications Compared to Pelzl, Šimka, et al

Internal vs. External Control

Pelzl, Šimka
- ARM7
- FPGA

New
- host
- control
- ECM units

Memory Management

Pelzl, Šimka
- suboptimal use of memory space

New
- bit tables & consolidation of memory resources

Functional Units

Pelzl, Šimka
- MUL
- ADD/SUB

New
- MUL1
- MUL2
- ADD/SUB

Montgomery Multiplier

Pelzl, Šimka
- Based on
  - Tenca, Koc
  - CHES 1999
  - IEEE Trans. Comp. 2003
  - word-based CPA and/or CSA

New
- Based on
  - McIvor, McLoone et al
  - Asilomar 2003
  - full-length CSAs
  - word-length CPAs
Comparison with the Proof-of-Concept Design by Pelzl & Šimka

Time x Area Product

Assuming the same memory management (i.e., improved memory management in Pelzl/Šimka):

Improvement

Phase 1 x 3.4
Phase 2 x 5.6
FPGAs vs Microprocessors

Execution Time

Pentium 4 Xeon 2.8 GHz

- Spartan 3 XC3S5000-5: 21.4 ms Phase 1, 24 ms Phase 2
- Virtex II XC2V6000-6: 14.2 ms Phase 1, 15.9 ms Phase 2
- Test program (No optimizations): 18.3 ms Phase 1, 18.6 ms Phase 2
- GMP-ECM: Phase 1 Optimizations off: 13.5 ms Phase 1, 13.5 ms Phase 2
- GMP-ECM all optimizations on: 11.3 ms Phase 1, 13.5 ms Phase 2
FPGAs vs Microprocessors

Number of ECM Computations per Second

<table>
<thead>
<tr>
<th></th>
<th>Number of Computations per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtex II</td>
<td>430</td>
</tr>
<tr>
<td>XC2V6000-6</td>
<td></td>
</tr>
<tr>
<td>Spartan 3</td>
<td>287</td>
</tr>
<tr>
<td>XC3S5000-5</td>
<td></td>
</tr>
<tr>
<td>Test program (No optimizations)</td>
<td>27</td>
</tr>
<tr>
<td>GMP-ECM: Phase 1 optimizations off</td>
<td>37</td>
</tr>
<tr>
<td>GMP-ECM all optimizations on</td>
<td>40</td>
</tr>
</tbody>
</table>

Pentium 4 Xeon 2.8 GHz

10.7 x 7.2 x .67 x .92 x

Hardware Implementation of the Elliptic Curve Method of Factoring
ASIC Results

- Synthesized Single ECM unit using TSMC 90nm library
- Maximum frequency achieved 261 MHz
- Total Area Requirement 954,567 au
**ASICs vs FPGAs**

- 2–3 times improvement in clock frequency (~260 MHz vs. 80-120 MHz)
- 5–10 times improvement in circuit area

but about $1,000,000 of one-time non-recurring costs needed for the back-end design & preparation of masks for fabrication
Summary

- Designed a novel architecture for ECM with two multipliers and one adder/subtractor per ECM unit
- Selected a 5 times more efficient multiplier compared to the design Pelzl, Šimka, et al
- Designed and implemented an original hardwired control unit for Phase 1 and Phase 2 of ECM composed of 18 state machines
Summary

- Verified the VHDL code through functional and timing simulation by comparison with the operation of test software implementation written in C and an optimized public domain software implementation GMP-ECM.
- Demonstrated the speed up by a factor 9.3 for Phase 1 and 15.0 for Phase 2 compared to the design by Pelzl, Šimka, et al.
Summary

- Ported the VHDL code to 5 different families of FPGA devices and to a standard-cell ASIC based on 90 nm TSMC library
Conclusions

- ECM running on low-cost FPGA devices, such as Spartan 3, outperformed high-performance devices, such as Virtex II, in terms of performance to cost ratio by a factor of 13.8.

- ECM running on low-cost FPGA device, Spartan 3, outperformed highly optimized software implementation of ECM, ECM-GMP, running on Pentium 4, by an order of magnitude in terms of the performance to cost ratio.
Conclusions

- ASIC implementations can compete with implementations based on low-cost FPGAs only for relatively high volumes (> 10,000 units) that compensate for the initial non-recurring costs
Publications related to ECM

Presented at Special purpose Hardware for Attacking Cryptographic System (SHARCS ’06)

April 03 – 04, 2006

Cologne, Germany
Publications related to ECM

Accepted as a paper at Workshop on Cryptographic Hardware and Embedded Systems (CHES 2006)
Thank you!

Questions???