Serial vs. Parallel Elliptic Curve Crypto Processor Designs

Adnan Abdul-Aziz Gutub
Umm Al-Qura University (UQU)

Abdul-Aziz Tabakh, Ayed Al-Qahtani, Alaaeldin Amin
King Fahd University of Petroleum & Minerals (KFUPM)

Outline

• Introduction
  • Crypto-systems
  • Elliptic curve cryptography (ECC)
• Aladdin Modulo Multiplier
• Architecture & Development
• Experiment results & Comparison
• Achievements & Conclusions
Introduction:

Secure Communication

Classic Cryptography

• Substitution (Caesar)
  - Transposition
  - Enigma Machine
    - Vigenere
    - Block (Hill)
• Vernam (one time pad)
  - DES
  - AES
Symmetric key algorithms

- Encryption and decryption keys are known to both parties.
- They are usually related (if not identical), making it easy to derive the decryption key once the encryption key is known.
- DES, AES (Rijndael).
- A secret must be known (agreed upon) to communicating parties so they can generate encryption and decryption keys.
- Key distribution and/or management problem.

Symmetric-key cryptography

- Encryption/decryption is performed using one secret key: e.g. DES and AES.
- Advantage: they are faster in execution
  - Straightforward transformations.
  - Can be pipelined to give better performance.
- Problem is to find efficient method to exchange keys securely: key distribution problem.
- If secret key is disclosed during key exchange >>> whole symmetric key crypto algorithm >>> completely vulnerable.
Public-key cryptography

- Public-key crypto algorithms (asymmetric cryptographic algorithms) e.g. RSA use separate public and private keys to perform encryption and decryption.
- The public key is made widely available and is used by communicating parties to encrypt data. Only the party with the correct corresponding private key can decrypt data.
- Public-key algorithms are the most secure cryptographic algorithms because they are based on an underlying mathematically hard-to-solve problem like integer factorization problem, discrete logarithm problem etc.
- They are also substantially slower than symmetric-key cryptography algorithms.
- Public key algorithms are commonly used in practice for the transport of keys subsequently used for bulk data encryption and decryption by symmetric-key algorithms and other applications.
Basic Cryptographic Applications

- **Confidentiality**  
  - Hiding contents of messages exchanged in a transaction

- **Authentication**  
  - Ensuring that the origin of a message is correctly identified

- **Integrity**  
  - Ensuring that only authorized parties are able to modify computer system assets and transmitted information

- **Non-repudiation**  
  - Requires that neither of the authorized parties deny the aspects of a valid transaction

Other Cryptographic Applications

- **Digital Signatures**  
  - Allows electronically sign (personalize) the electronic documents, messages and transactions

- **Identification**  
  - Is capable of replacing password-based identification methods with more powerful (secure) techniques

- **Key Establishment**  
  - To communicate a key to your correspondent (or perhaps actually mutually generate it with him) whom you have never physically met before

- **Secret Sharing**  
  - Distribute the parts of a secret to a group of people who can never exploit it individually
Other Cryptographic Applications

- **E-commerce**
  - carry out the secure transaction over an insecure channel like Internet

- **E-cash**
  - The cash can be sent securely through computer networks
  - The cash cannot be copied and reused
  - The spender of the cash can remain anonymous
  - The transaction can be done *offline*
  - The cash transferred to others
  - A piece of cash can be divided into smaller amounts

- **Games**
  - Flipping coins over the phone

- **Electronic Voting**

---

**Why Elliptic Curve Cryptography (ECC)?**

- Integer factorization problems (RSA)
- Discrete Logarithm problems (Diffie-Helman, ElGamal)
- Elliptic Curve Cryptosystems (ECC)

<table>
<thead>
<tr>
<th>Algorithm Family</th>
<th>Bit Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer Factorization (IF)</td>
<td>1024</td>
</tr>
<tr>
<td>Discrete Logarithm (DL)</td>
<td>1024</td>
</tr>
<tr>
<td>Elliptic Curve Crypto (ECC)</td>
<td>160</td>
</tr>
</tbody>
</table>
Elliptic curve cryptography (ECC)

- Elliptic curve in GF(P)
  - \( y^2 = x^3 + ax + b \).
- Enc/Dec:
  - Enc:
    - base point \((x, y)\). Public key \( P_A = n_A(x, y) \).
    - choose a random integer \( k \).
    - \( C_m = \{ k(x, y), (x_m, y_m) + kP \} \)
  - Dec:
    - \((x_m, y_m) + kP_A\) - \( n_A(k(x, y)) \)
    - \((x_m, y_m) + k(n_A(x, y)) - n_A(k(x, y))\)
    - \((x_m, y_m)\).
- Multiples of the base point, \((x, y)\) consumes most of the time.

\[
\begin{align*}
\text{Q+P} &= (x_1, y_1) + (x_2, y_2) = (x_3, y_3) \quad \text{where } x_1 \neq x_2 \\
\lambda &= (y_2 - y_1)/(x_2 - x_1) \\
x_3 &= \lambda^2 - x_1 - x_2 \\
y_3 &= \lambda(x_1 + x_2) - y_1
\end{align*}
\]

\[
\begin{align*}
\text{Q+Q} &= (x_1, y_1) + (x_1, y_1) = (x_3, y_3) \quad \text{where } x_1 \neq 0 \\
\lambda &= (3(x_1)^2 + a)/(2y_1) \\
x_3 &= \lambda^2 - 2x_1 \\
y_3 &= \lambda(x_1^2 + \lambda - 1) y_1
\end{align*}
\]

Inversion:

\[
\begin{align*}
\lambda &= (y_2 - y_1)/(x_2 - x_1) \\
x_3 &= \lambda^2 - x_1 - x_2 \\
y_3 &= \lambda(x_1 + x_2) - y_1
\end{align*}
\]
### Projective Coordinates

- Elimination of inversion $\Rightarrow$ Projective Coordinates.

**Add**

$$\begin{align*}
(x, y) &\mapsto (X/Z, Y/Z) \Rightarrow (X, Y, Z) \\
\lambda_2 &= XZ_3 \\
\lambda_3 &= X \lambda_2 - \lambda_3 \\
\lambda_4 &= Y_3 Z_3 \\
\lambda_5 &= Y_3 \lambda_1 \\
\lambda_6 &= \lambda_3 - \lambda_4 \\
\lambda_7 &= \lambda_2 + \lambda_2 \\
\lambda_8 &= \lambda_5^2 Z_1 Z_2 - \lambda_7 \\
Z_3 &= Z_2 Z_3 \lambda_5 \\
X_3 &= \lambda_6 \lambda_3 \\
Y_3 &= \lambda_6 X_3 Z_3 - \lambda_8 \\
\end{align*}$$

**Double**

$$\begin{align*}
(x, y) &\mapsto (X/Z, Y/Z) \Rightarrow (X, Y, Z) \\
\lambda_1 &= 3X_1^2 + nZ_1^2 \\
\lambda_2 &= Y_1 Z_1 \\
\lambda_3 &= X_1 Y_1 \lambda_2 \\
\lambda_4 &= \lambda_2^2 - 8 \lambda_3 \\
X_3 &= 2\lambda_4 \lambda_2 \\
Y_3 &= \lambda_4 (4\lambda_3 \lambda_4 - 8(Y_1 \lambda_2)^2) \\
Z_3 &= 8 \lambda_4^2 \\
\end{align*}$$

$$15M$$

---

### Aladdin Modulo Multiplier

- It is required to compute $P = A \cdot B \mod N$
  where $A, B, N$ are $n$-bits unsigned integers.

| I. Initialize | P ← θ  \\
|              | W ← A•N  \\
|              | IF W≥0 Then W ← W-N  \\
| II.          |  \\
| a-shift      |  \\
| b-Add        | For i=k-1 down to 0 do  \\
| c-Scale      | P ← 2P  \\
|              | =1 Then P ← P+b,A.  \\
|              | Else P ← P+b,W.  \\
|              | Case P+1P-k is  \\
|              | c.1 00: P ← P (no scaling).  \\
|              | c.2 01: P ← P+2N.  \\
|              | c.3 10: P ← P+2N.  \\
|              | c.4 11: P ← P (no scaling).  \\
|              | End Case  \\
| III. Correct | IF P ≥ N Then P ← P-N  \\
|              | Else while P<0 do P ← P+N  \\
|              | Return(P)  \\

---
Study

- ECC doubling and adding operations done:
  - in sequence.
  - In parallel.
- Performance, speed and area comparison.
- Show the gain obtained using VHDL.
- Use Aladdin modular multipliers.

Parallelization

Fig. 1 Adding two points data flow
Fig. 2 Doubling a point data flow graph
Architecture & Development

- VHDL is used to model the algorithms.
- Synthesize it of FPGA.
- Evaluate the speed and find out the space occupied.
  - Speed and space is the most important metrics for evaluating any digital design

```
Architecture & Development

- 10 registers are used to save the intermediate values
- 6 functional components are used in the parallel design
  - 1 Modulo Adder
  - 1 Modulo Shifter
  - 4 Modulo Multipliers
- Each register is associated with 8-by-1 multiplexer
- Each input of the functional components is associated with 16-by-1 multiplexer
```
### Registers contents – point adding

<table>
<thead>
<tr>
<th>R0</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>X1</td>
<td>Y1</td>
<td>Z1</td>
<td>X2</td>
<td>Y2</td>
<td>Z2</td>
<td>X2</td>
</tr>
<tr>
<td>mul 1</td>
<td>L4+Y1</td>
<td>Z2(R1,R6)</td>
<td>L5+Y2</td>
<td>Z1(R4,R2)</td>
<td>L1+X1</td>
<td>Z2(R0,R5)</td>
<td>L2+X2</td>
</tr>
<tr>
<td>add 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>L3=L2-L1</td>
<td></td>
</tr>
<tr>
<td>add 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>L6 Squ</td>
</tr>
<tr>
<td>mul 2</td>
<td>L3 squ</td>
<td>(R9,R9)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Z1 Z2</td>
</tr>
<tr>
<td>mul 3</td>
<td>L3Sq‘L7</td>
<td>(R0,R4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Z2 L3Cube</td>
</tr>
<tr>
<td>add 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>L8 ++</td>
<td>(R7-R4)</td>
</tr>
<tr>
<td>add 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>L9 ++(R6-R4)</td>
<td></td>
</tr>
<tr>
<td>mul 4</td>
<td>X3=L3L8</td>
<td>(R9,R4)</td>
<td>Z3=Z1Z2L3Cube(R2,R5)</td>
<td>Y1Z2L3Cube</td>
<td>(R1,R5)</td>
<td>L9L6</td>
<td>(R5,R3)</td>
</tr>
<tr>
<td>add 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y3+R5+R6</td>
<td></td>
</tr>
</tbody>
</table>

### Registers contents – point doubling

<table>
<thead>
<tr>
<th>R0</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td>a</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>shift 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2X (R0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>add 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3X (R0+R4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mul pre</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>aZ (R2,R3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mul 1</td>
<td>3X^2 (R0,R4)</td>
<td></td>
<td></td>
<td></td>
<td>L2=Y2</td>
<td>(R1,R2)</td>
<td>XY (R0,R1)</td>
</tr>
<tr>
<td>add 2</td>
<td>L1=3X^2*Z^2</td>
<td>(R0,R2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mul 2</td>
<td>YL2 (R1,R3)</td>
<td>L2^2 (R3,R0)</td>
<td></td>
<td></td>
<td>L3=XYL2</td>
<td>(R4,R3)</td>
<td>L1^2</td>
</tr>
<tr>
<td>shift 3</td>
<td></td>
<td></td>
<td></td>
<td>2L3 (R4)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>shift 4</td>
<td></td>
<td></td>
<td></td>
<td>4L3 (R5)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add 3</td>
<td></td>
<td></td>
<td></td>
<td>8L3 (R5)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>shift 5</td>
<td></td>
<td></td>
<td></td>
<td>L4-L1^2*8L3 (R6+R7)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>shift 6</td>
<td></td>
<td></td>
<td></td>
<td>4L2 (R3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>shift 7</td>
<td></td>
<td></td>
<td></td>
<td>8L2 (R6)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mul 3</td>
<td>(YL2)^2 (R1,R1)</td>
<td></td>
<td></td>
<td></td>
<td>L2L3^2</td>
<td>(R3,R2)</td>
<td>L1 L3</td>
</tr>
<tr>
<td>shift 8</td>
<td></td>
<td></td>
<td></td>
<td>2L4L2 (R6,R7)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>shift 9</td>
<td></td>
<td></td>
<td></td>
<td>4L1L2^2 (R1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>shift 10</td>
<td></td>
<td></td>
<td></td>
<td>8/L2L2^2 (R1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add 5</td>
<td>L1L5 - 8L1L2^2 (R3-R1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Experiment results & Comparison

- The maximum clock frequency 20.547MHz
  \[ \Rightarrow \text{the maximum clock period is 48.669ns.} \]
- The total size of the design is equivalent to 394,472 gates.
  - **Data Path** = 344,876 gates
  - **Control Unit** = 1,419 gates (<1%)
  - **Buffering** = 48,177 gates

<table>
<thead>
<tr>
<th>Component</th>
<th>Gate count per Component</th>
<th>Serial Design</th>
<th>Parallel Design</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Count</td>
<td>Total gate count</td>
</tr>
<tr>
<td>Multiplier</td>
<td>36,279</td>
<td>1</td>
<td>36,279</td>
</tr>
<tr>
<td>Adder</td>
<td>9,570</td>
<td>1</td>
<td>9,570</td>
</tr>
<tr>
<td>Shifter</td>
<td>2,862</td>
<td>1</td>
<td>2,862</td>
</tr>
<tr>
<td>Register</td>
<td>1,283</td>
<td>8</td>
<td>10,264</td>
</tr>
<tr>
<td>Mux 4x1</td>
<td>2,400</td>
<td>8</td>
<td>19,200</td>
</tr>
<tr>
<td>Mux 8x1</td>
<td>5,286</td>
<td>5</td>
<td>26,430</td>
</tr>
<tr>
<td>Mux 16x1</td>
<td>11,058</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Total Count of Gates</td>
<td></td>
<td>104,605</td>
<td>344,876</td>
</tr>
</tbody>
</table>

- **Multipliers** occupy the largest portion of the design (almost 35% is serial design and 42% is the parallel design)
- **Portion of designs** = parallel design is 3.3 times larger than the serial design
Experiment results & Comparison

<table>
<thead>
<tr>
<th>matrix</th>
<th>Serial Design</th>
<th>Parallel Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>2,039,797.5</td>
<td>1,724,380</td>
</tr>
<tr>
<td>AT^2</td>
<td>39,776,051.25</td>
<td>8,621,900</td>
</tr>
<tr>
<td>A^2T</td>
<td>2.13373×10^{11}</td>
<td>5.94697×10^{11}</td>
</tr>
</tbody>
</table>

- Parallel design is better in most cases.
- Serial is better when area is very important.

Achievements & Conclusions

- We implement Aladdin modulo multiplier
- We investigate the cost (Area & Time) tradeoffs for serial and parallel multiplier for ECC possessor.
- Exploiting parallelism boost up performance
- Parallel is better in most cases
  - excluding when hardware area is important
Serial vs. Parallel Elliptic Curve Crypto Processor Designs

Adnan Abdul-Aziz Gutub
Umm Al-Qura University (UQU)

Abdul-Aziz Tabakh, Ayed Al-Qahtani, Alaaeldin Amin
King Fahd University of Petroleum & Minerals (KFUPM)

IADIS International Conference Applied Computing 2013
Fort Worth, Texas, USA, 22-25 October 2013