Modular Exponentiation using Parallel Multipliers

S.H. Tang, K.S. Tsui and P.H.W. Leong
Custom Computing Laboratory
The Chinese University of Hong Kong
http://www.cse.cuhk.edu.hk/~phwl

Overview

- Introduction
- Algorithms
- Architecture
- Results
- Conclusion
FPGAs for Cryptography

- Particularly suitable for cryptographic accelerators
- State of the art VLSI performance
  - Faster products with zero design effort
- Multiple algorithms on the same hardware
  - Change functionality by downloading different designs to the same device
- Faster design times
  - More sophisticated algorithms, lower design cost, shorter time to market
- No NRE costs
  - Cheaper for low volumes
  - Conversion to ASIC for high performance, high volume
- Only recently have become cost effective for cryptography

This work

- High radix implementation of modular exponentiation
  - Utilizes dedicated multipliers of Virtex II devices
  - Systolic design
  - High radix
Overview

- Introduction
- Algorithms
- Architecture
- Results
- Conclusion

RSA Algorithm

- Invented in 1977 by Rivest, Shamir and Adleman
- Credit card machines, bank ATMs, e-mail applications, web browsers and mobile phones
- Encryption, key exchange, authentication
RSA Example

- $P=11, Q=7$ (secret)
- $N=pq=77$ & $E=13$ (public)
- Anyone can encrypt a message e.g. $M=4$
  - $4^{13} \mod 77 = 53$
- Decryption
  - $(P-1)(Q-1)=60$, $13 \times 37 \mod 60 = 1$ ($D=37$)
  - $53^{37} \mod 77 = 6283580383668332248635694548393830494073$
    $1973668146791149026213 \mod 77 = 4$

Modular Exponentiation Algorithm ($P=C^E$)

- L-R binary exponentiation method
- On average takes 1.5h multiplications
- Multiplications are modulo $M$

```
P = 1;
for i = h-1 .. 0 do
{
    P = P x P;
    if $e_i = 1$
        P = P x C;
}
The Montgomery Algorithm

- Used in most software and hardware implementations of modular exponentiation
- Performed in an R-residue where R is chosen as a power of 2
  - Divisions are shift operations
  - Reduction is interleaved with the multiplication

Montgomery Multiplication

- Input: A, B and M’
- Output: S = ABR⁻¹ (mod M)
- Precompute M’ s.t. (-M m’) mod R = 1
- Note that R⁻¹ term is unwanted

\[
\text{for } i = 0 \ldots n \text{ do}
\{
\begin{align*}
q &= ((S \mod \beta) \times (M' \mod \beta)) \mod \beta \\
S &= (S + qM) / \beta + a_iB
\end{align*}
\}
\]
Orup’s optimization simplifies calc
of q and reduces logic levels

\[ S = \frac{(S + qM)}{\beta} + a_i B \]
\[ S = \frac{S}{\beta} + \frac{qM}{\beta} + z + a_i B \]

---

Optimized Montgomery
Multiplication (Orup)

- Input: A, B and M’
- Output: S = ABR^{-1} (mod M)

\[
\mathcal{M} = M \ (M' \ mod \ \beta) \\
\text{for } i = 0 \ldots \ n \ do \\
\{ \\
\qquad q = S \ mod \ \beta \\
\qquad z = 1 \ if \ q \neq 0 \ else \ 0 \\
\qquad S = \frac{S}{\beta} + (q \mathcal{M})/\beta + z + a_i B \\
\} 
\]
Exponentiation using Montgomery Multiplication

- Extra pre and post processing steps are needed to remove unwanted $R^{-1}$ term.

\[
\begin{align*}
C^2 & \leq (C')(C') \\
C^4 & \leq (C^2)(C^2) \\
C^5 & \leq (C^4)(C) \\
C^{10} & \leq (C^5)(C^5)
\end{align*}
\begin{align*}
CR & \leq (C)(R^2)R^{-1} \\
C^2R & \leq (CR)(CR)R^{-1} \\
C^4R & \leq (C^2R)(C^2R)R^{-1} \\
C^5R & \leq (C^4R)(CR)R^{-1} \\
C^{10}R & \leq (C^5R)(C^5R)R^{-1} \\
C^{10} & \leq (C^{10}R)(1)R^{-1}
\end{align*}

Overview

- Introduction
- Algorithms
- Architecture
- Results
- Conclusion
Systolic Design

- Multiplier is too large for qM and aB
- Use a systolic array structure
  - Project vertically (row)

Processing element

\[
\begin{align*}
\theta_j &= q \times m_j / \beta \\
\phi_j &= a \times b_j / \beta \\
(qm)_j &= \langle q \times m_j \rangle \beta + \theta_{j-1} \\
(ab)_j &= \langle a \times b_j \rangle \beta + \phi_{j-1} \\
s_j &= s_{j+1} + (qm)_{j+1} + (ab)_j
\end{align*}
\]
Carry Save Adder

- The carry propagates through all PEs
  - 1024b addition takes 70 ns
  - Fast carry broken into 4 segments
- Use carry save architecture
  - The carry is saved in individual PE and used in later cycles
  - Addition is 17 bits

PE Design

- Single multiplier per cell (time multiplexed)
- Two clock cycles for each PE to compute S (pipelined)
  - ab on even cycles
  - qm on odd cycles
- Multiplex inputs to reduce wiring
PE Block Diagram

Semi-Systolic Array

• Has broadcast signals but reduces latency
Exponentiation

Overview

- Introduction
- Algorithms
- Architecture
- Results
- Conclusion
Results

- Platform: XC2V3000-6, XC2V4000-6
- Design description: VHDL
- Can work w/ or w/o CRT for RSA
- Designs of 512-bit and 1024-bit key are implemented
- All designs use a radix of $2^{17}$

Clock Cycles

- Multiplication
  - $2n$ cycles ($n =$ number of digits)
- Exponentiation
  - $3h/2$ cycles ($h =$ number of exponent bits)
- Total
  - Approximately $3nh$ cycles (i.e. $O(h^2)$)
Clock Cycles

<table>
<thead>
<tr>
<th>Operation</th>
<th>Generalized</th>
<th>512b</th>
<th>1024b</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Modular Multiplication</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MP × MP</td>
<td>2(n + 5)</td>
<td>74</td>
<td>134</td>
</tr>
<tr>
<td><strong>Modular Exponentiation</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Montgomery pre/post-processing</td>
<td>2(n + 5) × 3</td>
<td>222</td>
<td>402</td>
</tr>
<tr>
<td>Exponentiation</td>
<td>2(n + 5) × (b − h + \frac{3b}{2})</td>
<td>59200</td>
<td>209844</td>
</tr>
</tbody>
</table>

**Input / Output**

| Input of \( R^2, M, A, E \)       | \( 4 \times (b/64) + 1 \) | 37   | 69    |
| Output of \( A^b \mod M \)        | \( (b/64) + 1 \)           | 10   | 18    |

Measured total clock cycles 59468 210333

\( b \) – number of gross bits, 544 and 1054 for \( r=17 \) and \( h=512, 1024 \) respectively

Results

- Operating frequency: 90MHz
- Resources
  - 14334 of 14336 slices
  - 62 of 96 multipliers
- Using CRT, 1024-bit RSA can be computed in 0.66ms
  - i.e. 1.5 Mbps throughput
  - 10 times faster than an Intel P4 1.7GHz (OpenSSL with all optimizations)
- Software verification using OpenSSL library routines and random data
Performance Comparison

<table>
<thead>
<tr>
<th>Author</th>
<th>Year</th>
<th>Technology</th>
<th>Speed (1024b decryption)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orup et al</td>
<td>1991</td>
<td>ASIC 0.6u</td>
<td>(512b) 5.5 ms</td>
</tr>
<tr>
<td>Shand et al</td>
<td>1993</td>
<td>16x XC3090</td>
<td>6.1 ms</td>
</tr>
<tr>
<td>Itoh et al</td>
<td>1999</td>
<td>TMS320C6201 DSP</td>
<td>11.7 ms</td>
</tr>
<tr>
<td>Blum et al</td>
<td>2001</td>
<td>XC40250XV-09</td>
<td>3.1 ms</td>
</tr>
<tr>
<td>OpenSSL</td>
<td>2003</td>
<td>P4 1.7 GHz</td>
<td>6.9 ms</td>
</tr>
<tr>
<td>THIS WORK</td>
<td>2003</td>
<td>XC2V4000-6</td>
<td>0.7 ms</td>
</tr>
</tbody>
</table>

Conclusion

- Systolic array for modular exponentiation
- High clock frequency
  - Aggressive pipelining (systolic design)
  - Carry save adder
- Low number of clock cycles (approx 3nh)
  - Orup’s improved Montgomery Multiplication algorithm
  - 17x17 multipliers used to achieve high radix ($\beta=2^{17}$)
  - Reduced latency
- Parallelism
  - Multipliers and adders are used every clock cycle
- Performance of 1.5 Mb/s achieved on an XC2V4000-6 device