FPGA and ASIC Implementation of Rho and P-1 Methods of Factoring

Master’s Thesis Presentation
Ramakrishna Bachimanchi
Director: Dr. Kris Gaj
Contents

• Introduction
• Background
• Hardware Architecture
• FPGA and ASIC Design Flow
• Results
• Conclusions
RSA

In 1977

Ron Rivest, Adi Shamir & Leonard Adleman
developed the first public key cryptosystems, they called RSA
RSA

Public key \{e, N\}

Private key \{d, P, Q\}

Alice

\[ N = P \cdot Q \]
\( P, Q \) - large prime factors

Bob

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

Secure WWW, SSL

Browser → Network → WebServer

S/MIME, PGP

Alice → [Envelope] → Bob
Recommended key sizes for RSA

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

Old standard:

- Individual users
  - Short-term use (up to 2010): 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

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

Polynomial Selection

Relation Collection

Sieving

Relation Collection

200 bit & 350 bit numbers

Mini factoring

Pollard rho
p-1 method
ECM

Linear Algebra

Square Root
Rho Algorithm
Pollard’s Rho Method

Birthday paradox: If more than 23 “random” people are in a room (or even if they aren't) there is a more than 50% probability that the birthdays of two of them fall on the same day of the year.
Pollard's rho method - Example

$$N = 97 \cdot 1889 = 183\,233$$

$$x_{i+1} = x_i^2 + 1 \text{ mod } N$$

$$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4 \rightarrow x_5 \rightarrow x_6 \rightarrow x_7 \rightarrow x_8 \rightarrow x_9 \ldots$$

$$2 \rightarrow 5 \rightarrow 26 \rightarrow 677 \rightarrow 91864 \rightarrow 15449 \rightarrow 102236 \rightarrow 39678 \rightarrow 5749 \rightarrow 69062 \ldots$$

mod 97:

$$2 \rightarrow 5 \rightarrow 26 \rightarrow 95 \rightarrow 5 \rightarrow 26 \rightarrow 95 \rightarrow 5 \rightarrow 26 \rightarrow 95 \ldots$$

$$x_1 \equiv x_4 \equiv x_7 \equiv \ldots \text{ mod } q$$

$$x_2 \equiv x_5 \equiv x_8 \equiv \ldots \text{ mod } q$$

$$x_3 \equiv x_6 \equiv x_9 \equiv \ldots \text{ mod } q$$

$$x_1 \equiv x_4 \text{ mod } q$$

$$q \mid (x_1 - x_4)$$

$$q \mid N$$

$$q \mid \gcd(x_1 - x_4, N)$$

$$q = \gcd(-91\,859, 183\,233) = 97$$
Pollard’s Rho Method

\[
x_s \equiv x_e \mod q \\
x_{s+1} \equiv x_{e+1} \mod q \\
\ldots \\
x_{s+k} \equiv x_{e+k} \mod q
\]

\[\text{period} = e - s\]
Rho Algorithm- Floyd's Version

Initialize \( b = c = x_0 \)

1. choose the polynomial as \( f(x) = x^2 + a \)
2. calculate \( b = f(f(b)) \mod n \) and \( c = f(f(c)) \mod n \)
3. compute \( d = \gcd(b - c, n) \)
4. if \( 1 < d < n \), a non trivial factor of \( n \) is found
5. if \( d = 1 \) go to step 2
   
   if \( d = N \) change \( a \) and go to step 1
Rho Method - Floyd's Version

\[
\begin{align*}
X_1 - X_2 & \quad X_1 - X_3 & \quad X_1 - X_4 & \quad X_1 - X_5 & \quad X_1 - X_6 & \quad \cdots & \quad X_1 - X_i \\
X_2 - X_3 & \quad X_2 - X_4 & \quad X_2 - X_5 & \quad X_2 - X_6 & \quad X_2 - X_7 & \quad \cdots & \quad X_2 - X_i \\
X_3 - X_4 & \quad X_3 - X_5 & \quad X_3 - X_6 & \quad X_3 - X_7 & \quad X_3 - X_8 & \quad \cdots & \quad X_3 - X_i \\
X_4 - X_5 & \quad X_4 - X_6 & \quad X_4 - X_7 & \quad X_4 - X_8 & \quad X_4 - X_9 & \quad \cdots & \quad X_4 - X_i \\
X_5 - X_6 & \quad X_5 - X_7 & \quad X_5 - X_8 & \quad X_5 - X_9 & \quad X_5 - X_{10} & \quad \cdots & \quad X_5 - X_i \\
X_6 - X_7 & \quad X_6 - X_8 & \quad X_6 - X_9 & \quad X_6 - X_{10} & \quad X_6 - X_{11} & \quad X_6 - X_{12} & \quad \cdots & \quad X_6 - X_i \\
X_7 - X_8 & \quad X_7 - X_9 & \quad X_7 - X_{10} & \quad X_7 - X_{11} & \quad X_7 - X_{12} & \quad X_7 - X_{13} & \quad X_7 - X_{14} & \quad \cdots & \quad X_7 - X_i \\
X_8 - X_9 & \quad X_8 - X_{10} & \quad X_8 - X_{11} & \quad X_8 - X_{12} & \quad X_8 - X_{13} & \quad X_8 - X_{14} & \quad X_8 - X_{15} & \quad X_8 - X_{16} & \quad \cdots & \quad X_8 - X_i \\
& \quad \cdots & \quad \cdots & \quad \cdots & \quad \cdots & \quad \cdots & \quad \cdots & \quad \cdots & \quad \cdots & \quad \cdots \\
X_k - X_{k+1} & \quad X_k - X_{k+2} & \quad X_k - X_{k+3} & \quad \cdots & \quad X_{2k} & \quad \cdots & \quad X_k - X_i
\end{align*}
\]
Pollard's Rho Algorithm - Floyd's Version

\[
f(x) = x^2 + a \text{ with } a \not\in \{-2, 0\}
\]

\# iterations \( t < 100 \sqrt{q_{\text{max}}} \) (\( q_{\text{max}} \) is the maximum factor we expect to find using rho method)

We choose random \( x_0 \) in the range \((0, N-1)\) and \( x_1 = f(x_0) \)

\[
\begin{align*}
V_2 & \quad V_1 \\
x_0 & \quad x_1 \\
\downarrow & \quad \downarrow \\
x_2 & \quad x_1 \\
\downarrow & \quad \downarrow \\
x_3 & \quad x_2 \\
\downarrow & \quad \downarrow \\
x_4 & \quad x_3 \\
\downarrow & \quad \downarrow \\
x_5 & \quad x_4 \\
\downarrow & \quad \downarrow \\
x_6 & \quad x_5 \\
\downarrow & \quad \downarrow \\
x_t & \quad x_{t/2} \\
\downarrow & \quad \downarrow \\
x_{t+2} & \quad x_{(t+2)/2} \\
\downarrow & \quad \downarrow \\
x_{2i} & \quad x_i \\
\downarrow & \quad \downarrow \\
x_{2(i+1)} & \quad x_{i+1} \\
\downarrow & \quad \downarrow \\
x_{2t} & \quad x_t \\
\end{align*}
\]

\[x_{2i+2} = f(f(x_{2i})), x_{i+1} = f(x_i)\]

\[d = \gcd(d, N)\]

Minimization for area and/or memory
Rho Algorithm- Floyd's Version Contd.

Inputs: \( x_0, a, f(x) = x^2 + a, N, t \) (even, \( > 2 \))

Outputs: \( q \) (such that \( q \mid N \))

\( v_1 = x_1 = f(x_0), v_2 = x_2 = f(x_1), temp = v_2 - v_1 = x_2 - x_1, d = 1 \)

for \( i = 2; i \leq t; i ++ \)

\{
\begin{align*}
v_2 &= v_1^2 \\
v_2 &= v_2 + a \\
v_2 &= v_2^2 \\
v_2 &= v_2 + a \\
v_1 &= v_1^2 \\
v_1 &= v_1 + a \\
temp &= v_2 - v_1 \\
d &= d * temp
\end{align*}
\}

\*all operations are done

\( q = \gcd (d, N) \)
Rho Method - Brent's Version

\[
\begin{align*}
X_1 \cdot X_2 & \quad X_1 \cdot X_3 & \quad X_1 \cdot X_4 & \quad X_1 \cdot X_5 & \quad X_1 \cdot X_6 & \quad \cdots & \quad X_1 \cdot X_i \\
X_2 \cdot X_3 & \quad X_2 \cdot X_4 & \quad X_2 \cdot X_5 & \quad X_2 \cdot X_6 & \quad X_2 \cdot X_7 & \quad \cdots & \quad X_2 \cdot X_i \\
X_3 \cdot X_4 & \quad X_3 \cdot X_5 & \quad X_3 \cdot X_6 & \quad X_3 \cdot X_7 & \quad X_3 \cdot X_8 & \quad \cdots & \quad X_3 \cdot X_i \\
X_4 \cdot X_5 & \quad X_4 \cdot X_6 & \quad X_4 \cdot X_7 & \quad X_4 \cdot X_8 & \quad X_4 \cdot X_9 & \quad \cdots & \quad X_4 \cdot X_i \\
X_5 \cdot X_6 & \quad X_5 \cdot X_7 & \quad X_5 \cdot X_8 & \quad X_5 \cdot X_9 & \quad X_5 \cdot X_{10} & \quad \cdots & \quad X_5 \cdot X_i \\
X_6 \cdot X_7 & \quad X_6 \cdot X_8 & \quad X_6 \cdot X_9 & \quad X_6 \cdot X_{10} & \quad X_6 \cdot X_{11} & \quad \cdots & \quad X_6 \cdot X_i \\
X_7 \cdot X_8 & \quad X_7 \cdot X_9 & \quad X_7 \cdot X_{10} & \quad X_7 \cdot X_{11} & \quad X_7 \cdot X_{12} & \quad \cdots & \quad X_7 \cdot X_i \\
X_8 \cdot X_9 & \quad X_8 \cdot X_{10} & \quad X_8 \cdot X_{11} & \quad X_8 \cdot X_{12} & \quad X_8 \cdot X_{13} & \quad \cdots & \quad X_8 \cdot X_i \\
\cdots & & \cdots & & \cdots & & \cdots \\
X_k \cdot X_{k+1} & \quad X_k \cdot X_{k+2} & \quad X_k \cdot X_{k+3} & \quad \cdots & \quad X_2^k \cdot X_2^{k+1} & \quad \cdots & \quad X_2^k \cdot X_2^{k+1}
\end{align*}
\]
Rho Method - Brent’s Version
Sequence of Operations

\[ v_2 \]
\[ x_2 \]
\[ x_3 \]
\[ x_4 \]
\[ x_5 \]
\[ x_6 \]
\[ x_7 \]
\[ x_8 \]
\[ x_9 \]
\[ x_{10} \]
\[ x_{11} \]
\[ x_{12} \]
\[ x_{13} \]
\[ x_{14} \]
\[ x_{15} \]
\[ x_{16} \]

\[ d = 1 \]
\[ d = d \times (x_4 - x_2) \]
\[ d \times (x_7 - x_4) \]
\[ d \times (x_8 - x_4) \]
\[ d \times (x_{13} - x_8) \]

Minimization for execution time
24%
**Rho Algorithm - Brent’s Version**

*Inputs*: $x_0$, $a$, $f(x) = x^2 + a$, $N$, $t$ (even, $> 2$)

*Outputs*: $q$ (such that $q | N$)

$x_1 = f(x_0)$, $v_2 = v_1 = x_2 = f(x_1)$, $k = 1$

for $(i = 3; i \leq 2t; \; i++)$

{

$v_2 = f(v_2)$

if $(2^k + 2^{k-1} + 1 \leq i \leq 2^{k+1})$

{

$\text{temp} = v_2 - v_1$

$d = d * \text{temp}$

}

if $(i = 2^{k+1})$

{

$v_1 = v_2$

$k = k + 1$

}

}

$q = \gcd(d, N)$
p-1 Algorithm
**p-1 Algorithm**

- Based on Fermat’s Little Theorem
  - $a^{p-1} \equiv 1 \pmod{p}$
  - $a^{m(p-1)} \equiv 1 \pmod{p}$
  - $a^{m(p-1)} - 1 \equiv 0 \pmod{p}$

- $N$ – number to be factored
- $a$, any small integer
- $p$, non-trivial factor of $N$
  - Choose a small number $a$, such that $1 < a < N$
  - Choose a special number $k$
  - Compute $a^k \pmod{N} - 1$
  - Compute $\gcd(a^k \pmod{N} - 1, N)$
**p-1 algorithm**

**Inputs:**
- \(N\) – number to be factored
- \(a\) – arbitrary integer such that \(\gcd(a, N)=1\)
- \(B_1\) – smoothness bound for Phase 1
- \(B_2\) – smoothness bound for Phase 2

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

1: \( k \leftarrow \prod_{p_i} p_i^{e_i} \) such that \( p_i \) - consecutive primes \( \leq B_1 \)

\( e_i \) - largest exponent such that \( p_i^{e_i} \leq B_1 \)

2: \( q_0 \leftarrow a^k \mod N \)

3: \( q \leftarrow \gcd(q_0 - 1, N) \)

4: if \( q > 1 \)

5: return \( q \) (factor of \( N \))

6: else

7: go to Phase 2

8: end if

---

precomputations

main computations

postcomputations
### $p$-1 algorithm – Phase 2

09: \( d \leftarrow 1 \)

10: for each prime \( p = B_1 \) to \( B_2 \) do

11: \( d \leftarrow d \cdot (q_0^p - 1) \mod N \) \hspace{1cm} \textit{main computations}

12: end for

13: \( q \leftarrow \gcd(d, N) \)

14: if \( q > 1 \) then

15: return \( q \) \hspace{1cm} \textit{postcomputations}

16: else

17: return \( \text{FAIL} \)

18: end if
**p-1 Phase 1 – Numerical example**

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

\[ a = 2 \]

\[ B_1 = 20 \]

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

\[ q_0 = a^k \mod N = 2^{232\,792\,560} \mod 1\,740\,719 = 1\,003\,058 \]

\[ q = \gcd (1\,003\,058 - 1; 1\,740\,719) = 1361 \]

---

Why did the method work?

\[ q - 1 = 1360 = 2 \cdot 5 \cdot 17 \mid k \]

\[ a^k \mod q = a^{(q-1)m} \mod q = 1 \]

\[ q \mid a^{k-1} \]
Modular Exponentiation- Sliding Window Method

Input: \( g, e = (e_{t-t}, \ldots, e_1, e_0) \) with \( e_t = 1 \), and an integer \( w \geq 1 \)

Output: \( g^e \)

1. Precomputation

\[
\begin{align*}
g_1 &\leftarrow g, g_2 \leftarrow g^2 \\
\text{For } i \text{ from 1 to } (2^{w-1} - 1) \text{ do: } g_{2i+1} &\leftarrow g_{2i-1} \ast g_2
\end{align*}
\]

2. \( A \leftarrow 1, i \leftarrow t \)

3. While \( i \geq 0 \) do the following

\[\text{if } e_i = 0 \text{ then do: } A \leftarrow A^2, i \leftarrow i - 1\]

\[\text{otherwise (} e_i \neq 0\text{), find the longest bitstring } e_t e_{t-1} \ldots e_i \text{ such that } i - l + 1 \leq w\]

\[\text{and } e_i = 1, \text{ and do the following}\]

\[A \leftarrow A^{2^{i+1}} \ast g_{(e_t e_{t-1} \ldots e_i)}, i \leftarrow l - 1\]

4. Return(\( A \))
Sliding Window Method - Example

- calculating $g^{50}$, $e = (110010)_2$, window size 2
- Pre-computations $g^3$
- Main computations, $A \leftarrow 1$

  11 0010, window size = 2 and the value = “11” = 3
  $A \leftarrow (A)^4 \cdot g_3 = g^3$

  11 0 010
  $A \leftarrow A^2 = g^6$

  110 0 10
  $A \leftarrow A^2 = g^{12}$

  1100 1 0, window size = 1 and the value = ‘1’ = 1

  $A \leftarrow (A)^2 \cdot g_1 = g^{25}$

  11001 0
  $A \leftarrow A^2 = g^{50}$
Hardware Architecture
Top-level View

- FPGA / ASIC
- Control Unit
- I/O
- Global memory
- RAM
- Host computer
- Rho, p-1, unified Units
Low Level Arithmetic Units
Montgomery Multiplication

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

Original design
Global Memory - Rho

$\rho$ for unit 1

$\rho$ for unit 2

\ldots

\ldots

$\rho$ for unit $m$

$\alpha_0$

$\alpha$

$t$

Same for all units

No. of iterations
Local Memory - Rho
## Computation Flow

<table>
<thead>
<tr>
<th>MUL</th>
<th>ADD/SUB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 to 2t-1</td>
<td>(v_2 \leftarrow v_2^2)</td>
</tr>
<tr>
<td>cond1</td>
<td>(d \leftarrow d*\text{temp})</td>
</tr>
</tbody>
</table>

**cond1**: \(2^k + 2^{k-1} + 1 \leq i-1 \leq 2^{k+1}\)
Control Unit - Rho

- Memory Initialization
- Main Computations
- Reading Out Results
Global Memory – p-1

**Phase 1**

- \(N\) for unit 1
- \(N\) for unit 2
- \(\ldots\)
- \(N\) for unit \(m\)
- \(g_2\)
- \(g_1\)
- \(k_N\)
- \(k\)

**Initial values for All units**

**Phase 2**

- GCD_table[1]
- GCD_table[GMAXD]
- \(M_{\text{min}}\)
- \(M_{\text{max}}\)
- prime_table[1]
- prime_table[2]
- \(\ldots\)
- prime_table[PMAXD]

**Determined j such that**

\[1 \leq j \leq D\] and \(\gcd(j, D) = 1\)

**Determined m,j such that**

\(P = m.D - j\) is a prime
Local Memory – p-1

**Phase 1**

- \( N \)
- \( g_2 \)
- \( g_1 \)
- \( g_3 \)
- \( \vdots \)
- \( g_s \)
- \( *s = 2^k - 1 \)
- \( d = g^e \)

**Phase 2**

- \( N/d \)
- \( d^2 \)
- \( d \)
- \( d^{11} \)
- \( d^{13} \)
- \( \vdots \)
- \( d^{209} \)
- \( d^D \)
- \( d^{m.D} \)
- \( d^{m.D} - d^j \)
- \( x \)
Control Unit

Phase 1

- Memory Initialization
- Modular Exponentiation
- Reading Out Results

Phase 2

- Memory Initialization
- Pre-Computations
- Main-Computations
- Reading Out Results
Unified Architecture

ADD/SUB

Local Memory for p-1

Local Memory for Rho

Control Unit

Global Memory
Control Unit

- Memory Initialization
- Rho-Computations
- P-1 -Computations
- Reading Out Results
Control Unit

- Total 17 state machines with 140 states
  - 5 state machines with 45 states in Rho
  - 12 state machines with 103 states in P-1
- 5 Shift registers
- 9 Registers
- 13 Counters
- 22 Comparators

Original design
Design Flow
FPGA vs ASIC

- FPGA
  - Field Programmable Gate Array
  - Array of logic blocks
  - Switchable interconnect resources
  - Final user can set switches
  - Immediate use (“Zero” fab time)
  - Not good for high volume applications

- ASIC
  - Application Specific Integrated Circuit
  - Standard cells and Macros
  - Requires full manufacturing sequence
  - Good for high volume applications
FPGA Design Flow

Design Entry

- Specification
- RTL Description (VHDL / Verilog HDL)

Synthesis

Implementation

Configuration

Design Verification

- Functional Simulation
- Post-Synthesis Simulation
- Timing Simulation
- On Chip Testing
ASIC Design Flow

Front-End Design
- Synthesis
- Timing Analysis

Back-End Design
- Floorplanning
- Placement
- Clock Tree Synthesis
- Routing
- Design for Manufacturing

Design Analyzer
- Primetime

Astro
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
## FPGA Implementation of Single Units

<table>
<thead>
<tr>
<th>Results</th>
<th>Rho</th>
<th>P-1</th>
<th>Unified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resources</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-CLB Slices</td>
<td>1,680(4%)</td>
<td>1,749(5%)</td>
<td>2,042(6%)</td>
</tr>
<tr>
<td>-LUTs</td>
<td>2,714(4%)</td>
<td>2,875(4%)</td>
<td>3,451(5%)</td>
</tr>
<tr>
<td>-FFs</td>
<td>1,518(2%)</td>
<td>1,645(2%)</td>
<td>1,740(2%)</td>
</tr>
<tr>
<td>-BRAMs</td>
<td>0/144</td>
<td>2/144</td>
<td>2/144</td>
</tr>
<tr>
<td>Max. Clock Frequency</td>
<td>130 MHz</td>
<td>131 MHz</td>
<td>115 MHz</td>
</tr>
</tbody>
</table>

Target device is Virtex II XC2v6000-6
Number of unified units per FPGA

- Spartan 3 (XC3S5000-5) - Low-cost 19
- Virtex II (XC2V6000-6) - High-performance 21
- Spartan 3E (XC3S1600-5) - Low-cost 8
- Virtex 4 (XC4VLX200-11) - High-performance 42
Performance – Unified 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)

Operations per Second:
- Spartan 3: 581
- Virtex II: 819 \( \times 1.41 \)
- Spartan 3E: 290
- Virtex 4: 2,262 \( \times 7.8 \)
Performance to cost ratio

Unified Operations per second per $100

<table>
<thead>
<tr>
<th>Device</th>
<th>Performance</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spartan 3</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>XC3S5000-5</td>
<td>447</td>
<td>x 14.9</td>
</tr>
<tr>
<td>Spartan 3E</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>XC3S1600-5</td>
<td>828</td>
<td>x 11</td>
</tr>
<tr>
<td>Virtex II</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>XC2V6000-6</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Virtex 4</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>XC4VLX200-11</td>
<td>75</td>
<td></td>
</tr>
</tbody>
</table>
ASIC - Layout of p-1 - floorplanning
Layout of p-1 - placement
Layout of p-1 - clock tree synthesis
Layout of p-1 - Global Routing
Layout of p-1 - Detailed Routing
## Results - ASIC Implementation

<table>
<thead>
<tr>
<th>Operation</th>
<th>rho</th>
<th>p-1</th>
<th>Unified architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>1.15 mm$^2$</td>
<td>1.21 mm$^2$</td>
<td>1.8 mm$^2$</td>
</tr>
<tr>
<td>Max. Clock Frequency</td>
<td>200 MHz</td>
<td>200 MHz</td>
<td>200 MHz</td>
</tr>
<tr>
<td>Time for execution</td>
<td>3.52 ms</td>
<td>9.56 ms</td>
<td>13.1 ms</td>
</tr>
<tr>
<td># of operations per second (using maximum no. of units)</td>
<td>96,022</td>
<td>34,100</td>
<td>16,615</td>
</tr>
<tr>
<td>Core utilization ratio</td>
<td>70%</td>
<td>70%</td>
<td>65%</td>
</tr>
</tbody>
</table>

Area of Virtex II FPGA is 19.68 x 19.8 mm$^2$
( estimation by R.J. Lim Fong, MS Thesis, VPI, 2004)
FPGA vs ASIC - Area

Area of Virtex II FPGA is 19.68 x 19.8 mm²
(estimation by R.J. Lim Fong, MS Thesis, VPI, 2004)
ASIC 130 nm vs. Virtex II 6000 – \( \rho \) (20 units)

Area of Virtex II 6000
(estimation by R.J. Lim Fong, MS Thesis, VPI, 2004)

Area of an ASIC with equivalent functionality
ASICs vs. FPGAs

Source:
I. Kuon, J. Rose,
University of Toronto
“Measuring the Gap Between
FPGAs and ASICs”
IEEE Transactions on Computer-Aided
Design of Integrated Circuits and Systems,
<table>
<thead>
<tr>
<th>Name</th>
<th>Logic Only</th>
<th>Logic &amp; DSP</th>
<th>Logic &amp; Memory</th>
<th>Logic, Memory &amp; DSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>booth</td>
<td>33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rs_encoder</td>
<td>36</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cordic18</td>
<td>26</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cordic8</td>
<td>29</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>des_area</td>
<td>43</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>des_perf</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fir_restruct</td>
<td>34</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mac1</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aes192</td>
<td>49</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fir3</td>
<td>45</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>diffeq</td>
<td>44</td>
<td>13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>diffeq2</td>
<td>43</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>molecular</td>
<td>55</td>
<td>45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rs_decoder1</td>
<td>55</td>
<td>61</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rs_decoder2</td>
<td>48</td>
<td>43</td>
<td></td>
<td></td>
</tr>
<tr>
<td>atm</td>
<td></td>
<td></td>
<td>93</td>
<td></td>
</tr>
<tr>
<td>aes</td>
<td></td>
<td></td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>aes_inv</td>
<td></td>
<td></td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>ethernet</td>
<td></td>
<td></td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>serialproc</td>
<td></td>
<td></td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>fir24</td>
<td></td>
<td></td>
<td></td>
<td>9.8</td>
</tr>
<tr>
<td>pipe5proc</td>
<td></td>
<td></td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>raytracer</td>
<td></td>
<td></td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>Geomean</td>
<td>40</td>
<td>28</td>
<td>37</td>
<td>21</td>
</tr>
</tbody>
</table>
Table 3: Critical Path Delay Ratio (FPGA/ASIC) - Fastest Speed Grade

<table>
<thead>
<tr>
<th>Name</th>
<th>Logic</th>
<th>Logic &amp; Memory</th>
<th>Logic &amp; Memory</th>
<th>Logic, Memory &amp; DSP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Only</td>
<td>DSP</td>
<td>Memory</td>
<td>&amp; DSP</td>
</tr>
<tr>
<td>booth</td>
<td>4.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rs_encoder</td>
<td>3.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cordic18</td>
<td>3.6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cordic8</td>
<td>1.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>des_area</td>
<td>1.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>des_perf</td>
<td>2.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fir_restruct</td>
<td>3.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mac1</td>
<td>3.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aes192</td>
<td>4.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fir3</td>
<td>3.9</td>
<td>3.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>diffeq</td>
<td>4.0</td>
<td>4.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>diffeq2</td>
<td>3.9</td>
<td>4.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>molecular</td>
<td>4.4</td>
<td>4.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rs_decoder1</td>
<td>2.2</td>
<td>2.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rs_decoder2</td>
<td>2.0</td>
<td>2.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>atm</td>
<td></td>
<td></td>
<td>2.7</td>
<td></td>
</tr>
<tr>
<td>aes</td>
<td></td>
<td></td>
<td>3.7</td>
<td></td>
</tr>
<tr>
<td>aes_inv</td>
<td></td>
<td></td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td>ethernet</td>
<td></td>
<td></td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>serialproc</td>
<td></td>
<td></td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>fir24</td>
<td></td>
<td></td>
<td></td>
<td>2.5</td>
</tr>
<tr>
<td>pipe5proc</td>
<td></td>
<td></td>
<td></td>
<td>2.5</td>
</tr>
<tr>
<td>raytracer</td>
<td></td>
<td></td>
<td></td>
<td>1.4</td>
</tr>
<tr>
<td>Geomean</td>
<td>3.2</td>
<td>3.4</td>
<td>2.3</td>
<td>2.1</td>
</tr>
</tbody>
</table>
Contributions

- Verified the VHDL code through functional and timing simulation by comparison with the operation of test software implementation written in C.
- Ported the VHDL code to 4 different families of FPGA devices and to a standard-cell ASIC based on 130 nm TSMC library.
Conclusions

- 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 14.9
- ASIC Implementation outperforms FPGA with a factor of 50* in terms of area and 1.5 times in terms of frequency.

*In case of rho it is 50, for other architectures it may be less
Conclusions

- Low cost FPGA devices Spartan 3 and Spartan 3E are suitable for code-breaking.

- ASIC implementation is suitable when large number of chips (>1,000,000) are considered.
Future Work

- Implementation of Trial Division in Hardware
- Implementation of ECM in Hardware using one multiplier and one adder/subtractor
- Integrating Trial division, Rho, P-1 and ECM to build a co-factoring machine
- Experiments on COPACOBANA
Thank you!

Questions???