भारतीय प्रौद्योगिकी संस्थान खड़गपुर
Indian Institute of Technology
Kharagpur

High Speed Cryptoprocessor for η_T Pairing on 128-bit Secure Supersingular Elliptic Curves over Characteristic Two Fields

Santosh Ghosh, Dipanwita Roy Chowdhury, and Abhijit Das
Computer Science and Engineering
Indian Institute of Technology Kharagpur
India, 721302
{santosh,drc,abhij}@cse.iitkgp.ernet.in

Contents

- Introduction
- Contributions
  - Karatsuba multiplier over $F_{2^{1223}}$ field
  - Architectures for η_T pairing
- Conclusion

Introduction

- Pairing for designing cryptographic scheme introduced in 2000
  - ID-based encryption (IBE) scheme by Boneh and Franklin [1]
- It is well suited for identity based cryptography
  - has gained lot of importance in recent times
- As a natural consequence, implementations of pairings are also extremely important
- This paper broadly addresses design techniques of a pairing cryptoprocessor with high security level

[1] Boneh, D., and Franklin, M.K.: Identity-based encryption from the Weil pairing. Crypto 2001, LNCS 2139, pp. 213-229, 2001.

Introduction cont…

- Cryptographic pairings are computed on elliptic or hyperelliptic curves
  - defined over suitably large finite fields
  - having small embedding degree [2, 3].
- The security of a pairing depends on the underlying algebraic curves and respective field types
  - Example of an128-bit secure pairing :
    η_T pairing computed on a supersingular elliptic curve defined over $F_{2^{1223}}$ and having embedding degree k = 4.
- NIST recommendation: 128-bit symmetric security is essential beyond 2030
  - it is of importance to explore the efficient implementation techniques of 128-bit secure pairings on different platforms.

[2] Hoffstein, J., Pipher, J., and Silverman, J.H.: An introduction to mathmatical cryptography. Springer, 2008.
[3] Galbraith, S.: Pairings. In I. F. Blake, G. Seroussi, and N. P. Smart, editors. Advances in elliptic curve cryptography. London Mathematical Society Lecture Note Series, chapter IX. Cambridge University Press, 2005.

Existing works

- Hardware implementation of 128-bit secure pairings was introduced in 2009, individually by Kammler et al. [4] and Fan et al. [5].
  - Described hardware implementation techniques for computing 128-bit secure pairings over Barreto-Naehrig curves (BN curves) [6].
- Thereafter, designs in [7, 8, 9, 10] are appeared in literature
  - computes 128-bit secure pairings in 2.3ms, 16.4ms, 3.5ms, and 1.07ms.
- High-speed software implementations reported in [11, 12]
  - compute 128-bit secure pairings in 0.832ms and 1.87ms.

[4] Kammler et al: Designing an ASIP for cryptographic pairings over Barreto-Naehrig curves. CHES 2009.
[5] Fan et al: Faster Fp-arithmetic for cryptographic pairings on Barreto-Naehrig curves. CHES 2009.
[6] Barreto, P.S.L.M., and Naehrig, M.: Pairing-friendly elliptic curves of prime order. SAC 2005.
[7] Estibals, N.: Compact hardware for computing the Tate pairing over 128-bitsecurity supersingular curves. Pairing 2010.
[8] Ghosh et al: High speed flexible pairing cryptoprocessor on FPGA platform. Pairing 2010.
[9] Aranha et al: Optimal Eta pairing on supersingular genus-2 binary hyperelliptic curves. ePrint Report 2010/559.
[10] Duquesne et al: A FPGA pairing implementation using the residue number system. ePrint Report 2011/176.
[11] Beuchat et al: High-speed software implementation of the Optimal Ate pairing over Barreto-Naehrig curves. Pairing 2010.
[12] Beuchat et al: Multicore Implementation of the Tate Pairing over Supersingular Elliptic Curves. ePrint Report 2009/276.

Contributions

Major contributions of the paper are:

- It explores area-time tradeoff designs of Karatsuba multiplier over $F_{2^{1223}}$ field.
- It further explores high speed architecture for computing η_T pairing on supersingular elliptic curves.
- It provides the first hardware implementation result of an 128-bit secure pairing on elliptic curves over characteristic two fields.
- The proposed design achieves the fastest computation (190 µs) of an 128-bit secure pairing.

The $F_{2^{1223}}$-Multiplier

- Multiplication is the key operation of a pairing computation
- The 128-bit secure η_T pairing demands multiplication in an 1223-bit characteristic-two field.
- Karatsuba multiplication is an efficient and popular technique for fields like F_{q^m}.
  - It is a divide-and-conquer algorithm
  - An m-bit multiplication is divided recursively into several m/k-bit multiplications with small $k \in \{2, 3\}$.

The $F_{2^{1223}}$-Multiplier (Cont…)

- Multiplication for k = 2 in F_{q^m} could be computed as :

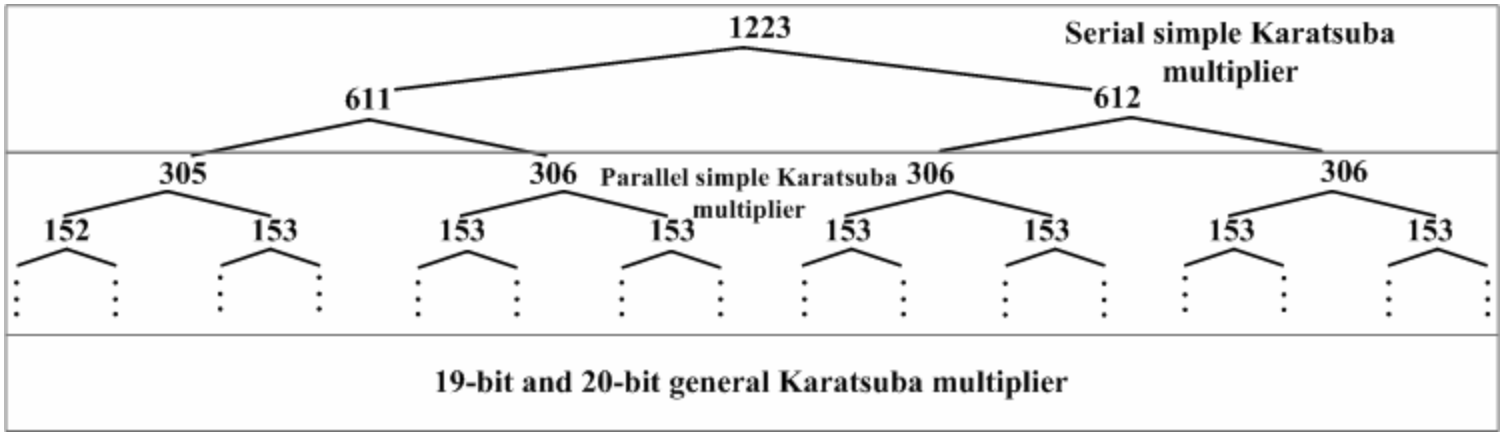
$$a \cdot b = (a_1 x^{\lceil m/2 \rceil} + a_0)(b_1 x^{\lceil m/2 \rceil} + b_0)$$
$$= a_1 b_1 x^m + [(a_1 + a_0)(b_1 + b_0) - a_1 b_1 - a_0 b_0]\, x^{\lceil m/2 \rceil} + a_0 b_0$$

- An m-bit multiplication can be performed by:
  - three m/2-bit multiplications
  - four m-bit and two m/2-bit additions.

- Implementation could be performed in **several ways.**
  - We show four different design techniques

1. Fully Parallel Multiplier for $F_{2^{1223}}$

- Decomposition is done as:



- The synthesis tool estimates 324342 LUTs for an 1223-bit **fully parallel** Karatsuba multiplier.
  - – makes it infeasible to implement on a single Virtex-4 FPGA device

2. Serial Use of 612-bit Parallel Multiplier

- It consists of a fully parallel 612-bit Karatsuba multiplier
  - Three 612-bit multiplications are performed in **serial** for computing a multiplication in $F_{2^{1223}}$
- The synthesis tool estimates 95324 LUTs.
  - It is feasible to implement on a high-end single FPGA device
  - a full pairing hardware demands much more circuits than a single multiplier
  - may infeasible to put in a single FPGA.

3. Serial Use of 306-bit Parallel Multiplier

- It consists of a fully parallel 306-bit Karatsuba multiplier
  - Nine 306-bit multiplications are performed in **serial** for computing a multiplication in $F_{2^{1223}}$
- The computation is performed as:

$$\begin{aligned} a \cdot b &= (a_1 x^{612} + a_0)(b_1 x^{612} + b_0) \\ &= a_1 b_1 x^{1222} + [(a_1 + a_0)(b_1 + b_0) - a_1 b_1 - a_0 b_0]\, x^{612} + a_0 b_0. \end{aligned}$$

- Three 612-bit multiplications: a_0b_0, a_1b_1, and $(a_1+a_0)(b_1+b_0)$
- performed as:

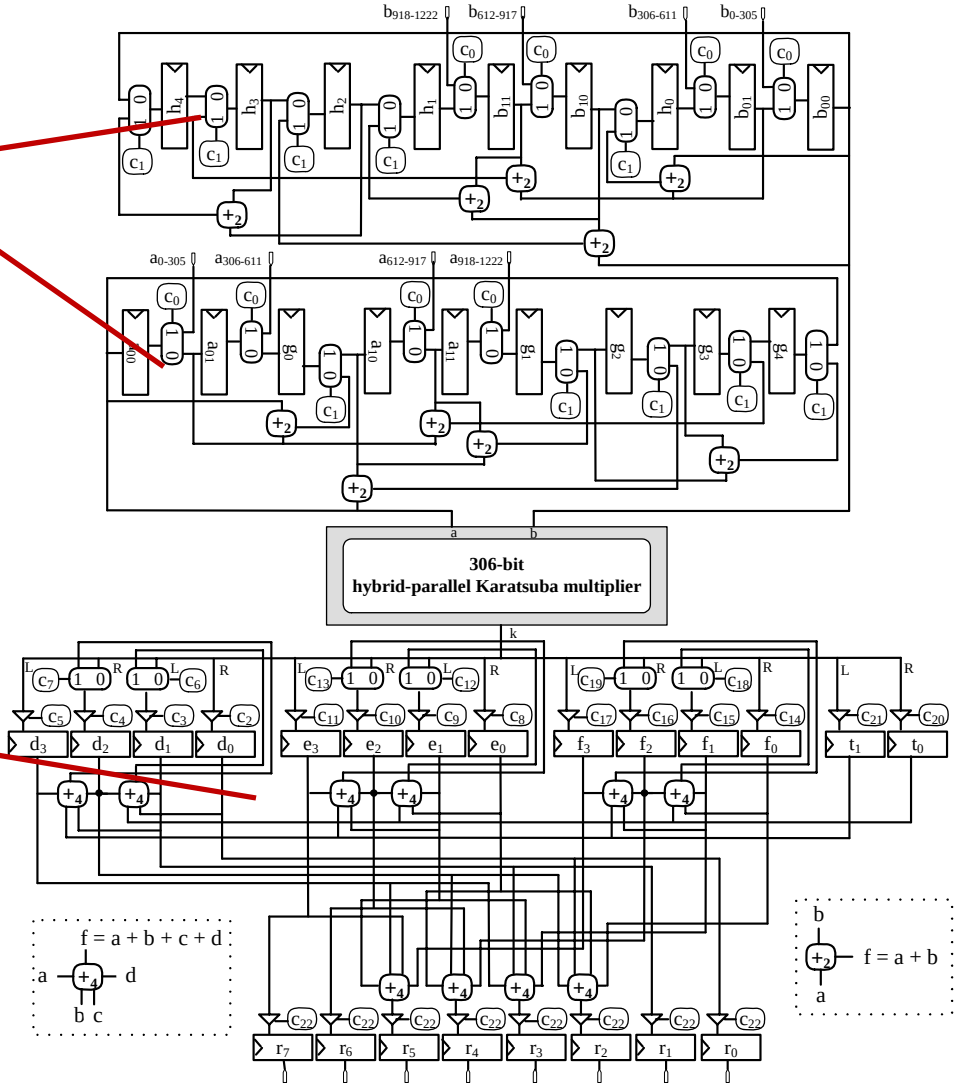
$$\begin{aligned} a_0 \cdot b_0 &= (a_{01} x^{306} + a_{00})(b_{01} x^{306} + b_{00}) \\ &= a_{01} b_{01} x^{612} + [(a_{01} + a_{00})(b_{01} + b_{00}) - a_{01} b_{01} - a_{00} b_{00}]\, x^{306} + a_{00} b_{00} \\ &= a_{01} b_{01} x^{612} + [g_0 h_0 - a_{01} b_{01} - a_{00} b_{00}]\, x^{306} + a_{00} b_{00}, \end{aligned} \quad (2)$$

Three 306-bit multiplications: $a_{01}b_{01}$, $a_{00}b_{00}$, and $(a_{01}+a_{00})(b_{01}+b_{00})$

- To sum up : nine 306-bit multiplications.

Multiplier Architecture

- The operands of nine multiplications are stored into two sets of nine 306-bit parallel shift registers.
- The registers are automatically reloaded by synchronous shift operations.
  - ensures two correct operands at a_{00} and b_{00} registers.
- Multiplier latency: one clock cycle.
- Partial results of 1223-bit multiplication are accumulated accordingly.
  (Algorithm is provided in Appendix)
- Latency of one 1223-bit multiplication: 10 clock cycles



Multiplier on FPGA

- Demands ≈30k LUT.
  - affordable to implement on a medium range FPGA

Multiplier type	**FPGA family**	**LUTs**	**Frequency** $[MHz]$	**Serial use**	**Multiplication latency** $[ns]$	$(A \cdot T)^{\S}$
Serial use of	Virtex-2	34 547	125	10	80.0	2.76
306-bit parallel	Virtex-4	34 325	168	10	60.0	2.06
multiplier	Virtex-6	30 148	250	10	40.0	1.21
§ : $(A \cdot T)$ represents product of *area* in LUTs and *time* in milliseconds.						

4. Serial use of 153-bit parallel multiplier

- Demands low resources : 16231 LUTs
- Requires 27 serial use
- Latency of one 1223-bit multiplication : 151ns
  - 2.5 times slower than 306-bit parallel multiplier
  - The $A \cdot T$ value is 2.46.
    - is 1.2 times higher than 306-bit parallel multiplier

Serial use of 306-bit parallel multiplier provides the most optimized design.

The η_T Pairing Cryptoprocessor over $F_{2^{1223}}$

- The pairing computation consists of two major operations
  1. the non-reduced pairing (Miller's algorithm)
  2. the final exponentiation

Main Features:

- Consists of a common datapath for both operations
- Adequate parallelism is applied to achieve high speed

Major difference with existing architecture of [13]:

- Contrast to two separate coprocessors current design has one processing unit for both operations.

[13] Beuchat et al.: Fast architectures for the T pairing over small-characteristic supersingular elliptic curves. IEEE Transactions on Computers,

Computation of η_T pairing

-1 and 2 are dependent operations

- Performed in serial

- Parallelism is applied inside these steps

Algorithm 1 : Computing the η_T pairing on $E/\mathbb{F}_{2^{1223}}$. Intermediate variables in uppercase belong to $\mathbb{F}_{(2^{1223})^4}$, whereas those in lowercase to $\mathbb{F}_{2^{1223}}$.

Input: $P(x_1, y_1)$ and $Q(x_2, y_2) \in E(\mathbb{F}_{2^{1223}})[r]$.
Output: $\eta_T(P, Q)$.

1. $x_1^{(0)} \leftarrow x_1$; $y_1^{(0)} \leftarrow y_1$; $x_2^{(0)} \leftarrow x_2$; $y_2^{(0)} \leftarrow y_2$;
2. $s^{(0)} \leftarrow x_1 + 1$;
3. $t_0^{(0)} \leftarrow s^{(0)} + x_2^{(0)}$; $t_1^{(0)} \leftarrow y_1^{(0)} + y_2^{(0)}$;
4. $f_0^{(0)} \leftarrow s^{(0)} \cdot t_0^{(0)} + t_1^{(0)}$; $f_1^{(0)} \leftarrow s^{(0)} + x_2^{(0)}$; $f_2^{(0)} \leftarrow 1$; $f_3^{(0)} \leftarrow 0$;
5. $F^{(0)} \leftarrow f_0^{(0)} + f_1^{(0)} u + f_2^{(0)} v + f_3^{(0)} uv$;
6. **for** i *from* 1 *to* 612 **do**
7. $s^{(i)} \leftarrow x_1^{(i-1)}$, $x_1^{(i)} \leftarrow \sqrt{x_1^{(i-1)}}$; $y_1^{(i)} \leftarrow \sqrt{y_1^{(i-1)}}$;
8. $t_0^{(i)} \leftarrow x_1^{(i)} + x_2^{(i-1)}$; $t_1^{(i)} \leftarrow y_1^{(i)} + y_2^{(i-1)} + x_1^{(i)} + 1$;
9. $g_0^{(i)} \leftarrow s^{(i)} \cdot t_0^{(i)} + t_1^{(i)}$; $g_1^{(i)} \leftarrow s^{(i)} + x_2^{(i-1)}$;
10. $G^{(i)} \leftarrow g_0^{(i)} + g_1^{(i)} u + v$;
11. $F^{(i)} \leftarrow F^{(i-1)} \cdot G^{(i)}$;
12. $x_2^{(i)} \leftarrow (x_2^{(i-1)})^2$; $y_2^{(i)} \leftarrow (y_2^{(i-1)})^2$;
13. **end for**
14. **return** $(F^{(612)})^{(2^{2446}-1)(2^{1223}-2^{612}+1)}$.

1

2

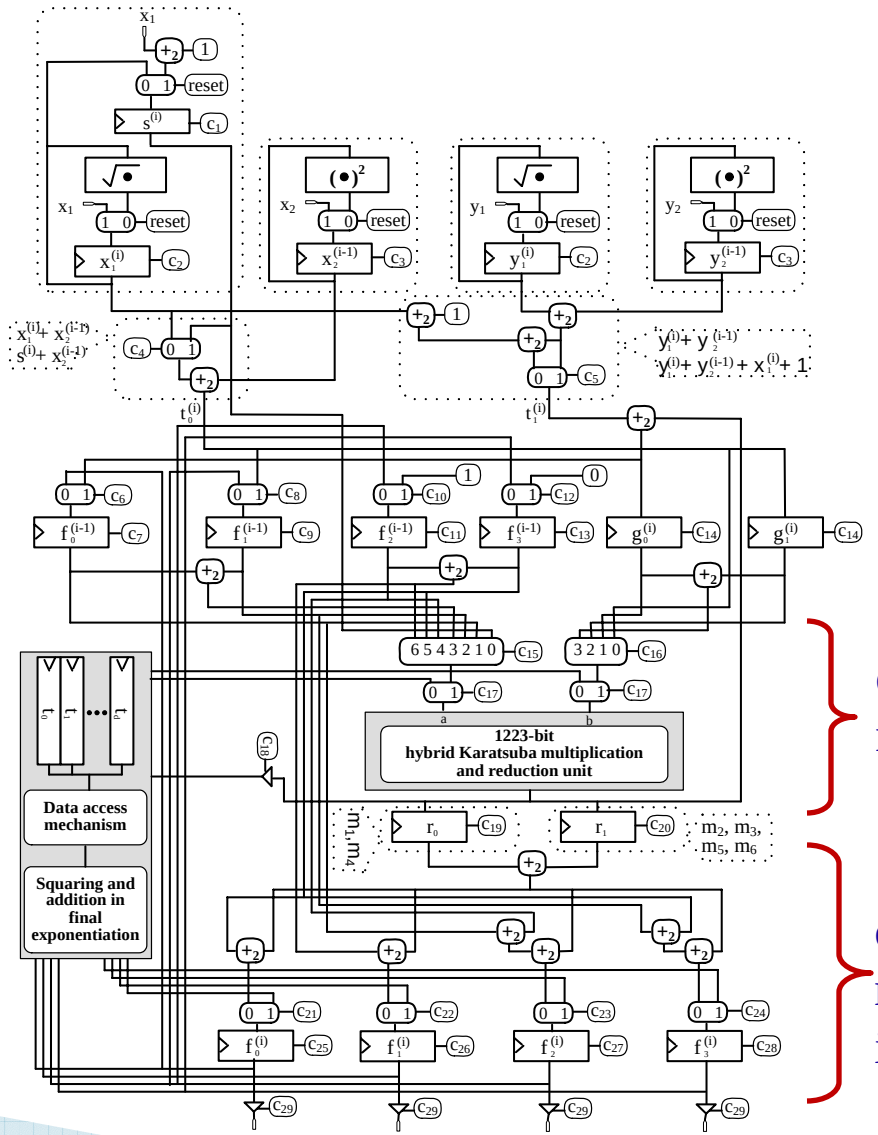
Final exponentiation

Proposed Architecture

compute squaring and square roots

hold intermediate results of (i-1)-th iteration.

performs operations other than multiplication in final exponentiation



computes multiplications

combine the results of *i*-th iteration

Execution procedure

- Initialization : Step 1 to step 5 is preformed synchronously with reset.

- Computation of $G^{(i)}$: - the sparse value of $G^{(i)} \in \mathbb{F}_{(2^{1223})^4}$ in {*1, u, v, uv*} basis is represented b $g_0^{(i)} + g_1^{(i)} u + 1$.
  - it consists of one multiplication $s^{(i)} \cdot t_0^{(i)}$.
  - in total it takes 12 clock cycles.

- Sparse Multiplication over $\mathbb{F}_{(2^{1223})^4}$: - it consists only six multiplications in $\mathbb{F}_{2^{1223}}$
  - in total it takes 61 clock cycles.

- Computation Cost of Miller's Algorithm :
  - one iteration takes 73 clock cycles.
  - in total its cost 44688 clock cycles.

Execution procedure

- Final Exponentiation : - the output of Miller's algorithm is raised to the power of $(2^{2446} - 1)(2^{1223} - 2^{612} + 1)$.

  - powering $G = g_0 + g_1u + g_2v + g_3uv$ $\mathbb{F}_{(2^{1223})^4}$ is easy :

$$G^{2^{1223}} = (g_0 + g_1 + g_2) + (g_1 + g_2 + g_3)u + (g_2 + g_3)v + g_3uv$$

  - further one inversion followed by one multiplication in $\mathbb{F}_{(2^{1223})^4}$.

  - in total, cost of final exponentiation is : *98M + 1842S + 135A.*

  - the proposed cryptoprocessor computes final exponentiation in 2922 clock cycles.

Total clock cycle count for computing an 128-bit secure η_T pairing is 47610 on our proposed cryptoprocessor.

Experimental results

- The whole design has been done in Verilog (HDL).
- Results from the place-and-route report of Xilinx ISE Design Suit is shown here:

Platform	**Slice**	**LUT**	**Frequency** $[MHz]$	**Clock Cycles**	**Security** $[bit]$	**Times** $[\mu s]$
Virtex-2	36534	69367	125			381
Virtex-4	35458	69367	168	47610	128	286
Virtex-6‡	15167	54681	250			190
‡ : One Virtex-6 slice consists of four LUTs and eight flip-flops.						

- it finishes computation of one 128-bit secure η_T pairing in *190 μs* on a Virtex-6 FPGA.

Comparison

Two aspects of the proposed design

1. Existing η_T pairing processors over characteristic-two fields

Designs	Curve	Security $[bit]$	FPGA	Area $[Slices]$	Frequency $[MHz]$	Times $[\mu s]$
Shu *et al.* [28]	$E/\mathbb{F}_{2^{557}}$	96	xc4vlx200-10	37931	66	675.5
Beuchat *et al.* [5]	$E/\mathbb{F}_{2^{691}}$	105	xc4vlx200-11	78874	130	18.8
This work	$E/\mathbb{F}_{2^{1223}}$	128	xc4vlx200-11	35458	168	286.0
This work	$E/\mathbb{F}_{2^{1223}}$	128	xc6vlx130t-3	15167	250	190.0

Higher security level

Lower area

Intermediate speed

[28] Shu et al. : Reconfigurable computing approach for tate pairing cryptosystems over binary fields. IEEE Transactions on Computers, 2009.
[5] Beuchat et al.: Fast architectures for the T pairing over small-characteristic supersingular elliptic curves. IEEE Transactions on Computers, 2011.

Comparison cont…

2. Existing 128-bit secure pairing implementations irrespective of underlying curve and field types.

Designs	**Curve**	**FPGA**	**Area**	**Freq.** $[MHz]$	**Times** $[\mu s]$	$A \cdot T$†
Duquesne et al. [9]§	$E/\mathbb{F}_{p256}$	Stratix III	4233 A‡	165	1070	-
Fan et al. [11]	$E/\mathbb{F}_{p256}$	xc6vlx240-3	4014 Slices, 42 DSP	210	1170	-
Kammler *et al.* [21]	$E/\mathbb{F}_{p256}$	$130nm$ CMOS	97000 Gates	338	15800	-
Fan *et al.* [12]	$E/\mathbb{F}_{p256}$	$130nm$ CMOS	183000 Gates	204	2900	-
Ghosh *et al.* [14]	$E/\mathbb{F}_{p256}$	xc4vlx200-12	52000 Slices	50	16400	852.8
Estibals [10]	$E/\mathbb{F}_{3^{5\cdot 97}}$	xc4vlx200-11	4755 Slices	192	2227	10.6
Aranha *et al.* [1]	$Co/\mathbb{F}_{2^{367}}$	xc4vlx25-11	4518 Slices	220	3518	15.9
This work	$E/\mathbb{F}_{2^{1223}}$	xc4vlx200-11	35458 Slices	168	286	10.1
This work	$E/\mathbb{F}_{2^{1223}}$	xc6vlx130t-3	15167 Slices	250	190	2.9

† $A \cdot T$ represents product of *area* in slices and *time* in seconds.
§ It provides 126-bit security instead of 128-bit.
‡ It has 8 Rowers, each consisting of two 36x36 DSP blocks and one 9x9 multiplier.

Intermediate area

Smallest AT value

- Highest speed
- First in micro-second range

Comparison cont…

Existing high speed software for 128-bit pairings are also slower than our proposed design. Here we summarize the software results.

Reference	Platform	Pairing	Curve	Frequency $[MHz]$	Times $[ms]$
Beuchat et al. [7]	core i7 2.8GHz	modified Tate	$E/\mathbb{F}_{3^{509}}$	2800	1.87
			$E/\mathbb{F}_{2^{1223}}$	2800	3.08
Naehrig et al. [26]	core2 Q6600	optimal-ate	$E/\mathbb{F}_{p256}$	2394	1.86
Beuchat et al. [6]	core i7 2.8GHz	optimal-ate	$E/\mathbb{F}_{p256}$	2800	0.83
Hankerson et al. [16]	64-bit core2	optimal-ate	$E/\mathbb{F}_{p256}$	2400	6.25
		η_T	$E/\mathbb{F}_{2^{1223}}$	2400	16.25
		η_T	$E/\mathbb{F}_{3^{509}}$	2400	13.75
Grabher et al. [15]	64-bit core2	ate	$E/\mathbb{F}_{p256}$	2400	6.01

The most efficient on take 0.83 ms, which is 4.3 times slower than our proposed design

Conclusion

- Area and time optimized hybrid Karatsuba multiplier for $F_{2^{1223}}$
  - Significant parallelism for speed up
  - Moderate area for optimization.

- Pairing cryptoprocessor
  - A common datapath for both non-reduced pairing and final exponentiation.
  - Reduces the overall logic cells
  - It computes η_T pairing in characteristic-two field with higher security (128:105) in **half area.**
  - it achieves **eight times speedup** and provides the **best *area * time*** product compared to the existing designs.

Minor corrections:

1.

Table 2. Computation of $F^{(i-1)} \cdot G^{(i)}$.

$m_1 : r_0 \leftarrow f_0^{(i-1)} \cdot g_0^{(i)}$;	$m_4 : r_0 \leftarrow f_2^{(i-1)} \cdot g_2^{(i)}$;
$m_2 : r_1 \leftarrow f_1^{(i-1)} \cdot g_1^{(i)}$;	$m_5 : r_1 \leftarrow f_3^{(i-1)} \cdot g_3^{(i)}$;

g_2 and g3 should be g_0 and g_1, respectively

2.

Table 6. Software for 128-bit secure pairings.

Reference	Platform	Pairing	Curve	Frequency [MHz]	Times [ms]
Beuchat et al. [7]	core i7 2.8GHz	modified Tate	$E/\mathbb{F}_{3^{509}}$	2800	1.87

[*MHz*] should be at the bottom of Frequency

Thank you