Architecture for a Fast Elliptic Curve Co-Processor

VLSI lab. Seminar

B. Ansari

Supervisor: Dr. H. Wu

March 2004

Outline

1. Introduction to Elliptic Curve Cryptography (From my first seminar)
2. Processor Architecture
3. Implementation
4. Result
5. Discussion

Elliptic Curve Cryptography (Why)

- ECC Proposed independently by Koblitz, Miller (1985)
- ECC Security is based on the elliptic curve discrete logarithm problem (ECDLP)
- It is estimated : RSA 4096 $\equiv$ ECC 313
- Implementation of elliptic curve cryptosystem requires:
  - Smaller chip size
  - Less power consumption
  - Faster (on Palm-Pilot), 512-bit RSA 3.4 min, 163-bit ECC-DSA 0.597 min
- IEEE, NIST, FIPS,

NIST guidelines for public key sizes for AES

ECC KEY SIZE (Bits)	RSA KEY SIZE (Bits)	KEY SIZE RATIO	AES KEY SIZE (Bits)
163	1024	1 : 6	
256	3072	1 : 12	128
384	7680	1 : 20	192
512	15 360	1 : 30	256

Supplied by NIST to ANSI X9F1

Elliptic Curve Cryptography (Application)

- Smartcards
- Handheld devices
- Wireless security
- IDs (Austrian National ID card)
- www.certicom.com 
- At University of Waterloo!

Elliptic curves over $GF(2^m)$

The Challenge

Curve	Field size (in bits)	Estimated number of machine days	Prize (US$)	Status
ECC2-79	79	352	HAC, Maple	SOLVED December 1997
ECC2-89	89	11278	HAC, Maple	SOLVED February 1998
ECC2K-95	97	8637	$ 5,000	SOLVED May 1998
ECC2-97	97	180448	$ 5,000	
ECC2K-108	109	1.3×10^{6}	$ 10,000	SOLVED April 2000
ECC2-109	109	2.1×10^{7}	$ 10,000	
ECC2K-130	131	2.7×10^{9}	$ 20,000	
ECC2-131	131	6.6×10^{10}	$ 20,000	
ECC2-163	163	2.9×10^{15}	$ 30,000	
ECC2K-163	163	4.6×10^{14}	$ 30,000	
ECC2-191	191	1.4×10^{20}	$ 40,000	
ECC2-238	239	3.0×10^{27}	$ 50,000	
ECC2K-238	239	1.3×10^{26}	$ 50,000	
ECC2-353	359	1.4×10^{45}	$ 100,000	
ECC2K-358	359	2.8×10^{44}	$ 100,000	

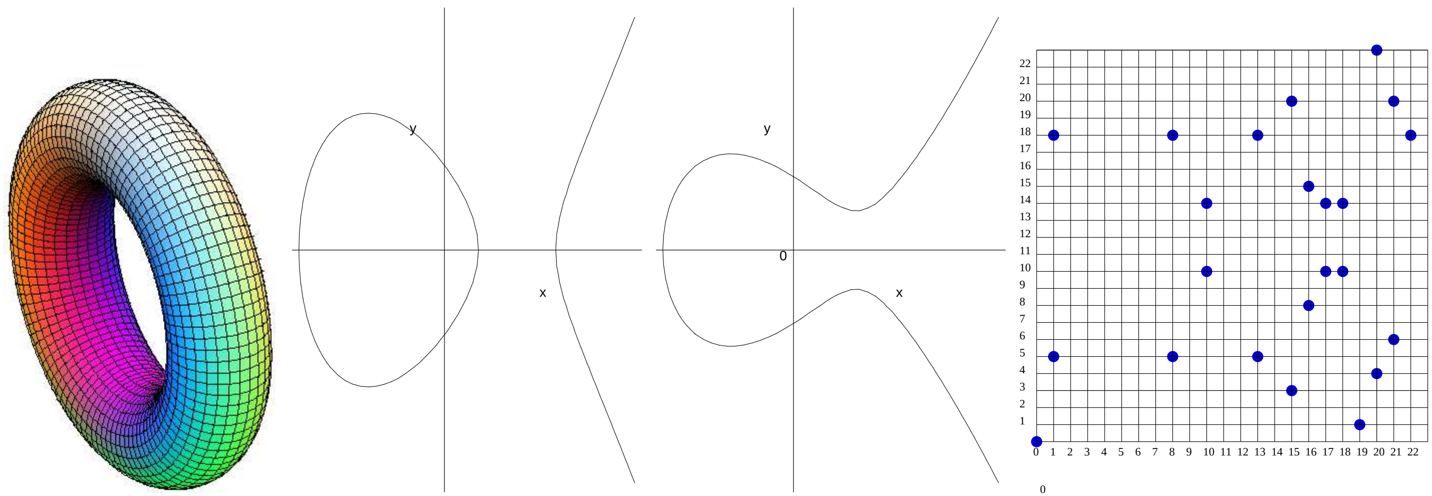
Elliptic Curve

- Elliptic curve E is the graph of an equation of the form

$$E : y^2 + a_1 xy + a_3 y = x^3 + a_2 x^2 + a_4 x + a_6 \tag{1}$$

where $\quad a_1, a_2, a_3, a_4, a_6 \in \mathcal{K}.$

- $\mathcal{K}$ is a *Field*. For example, Fields of $\mathbb{C}$, $\mathbb{R}$, $\mathbb{Q}$, Finite Field over prime $\mathbb{F}_p$ or Extension Field $\mathbb{F}_{p^n}$

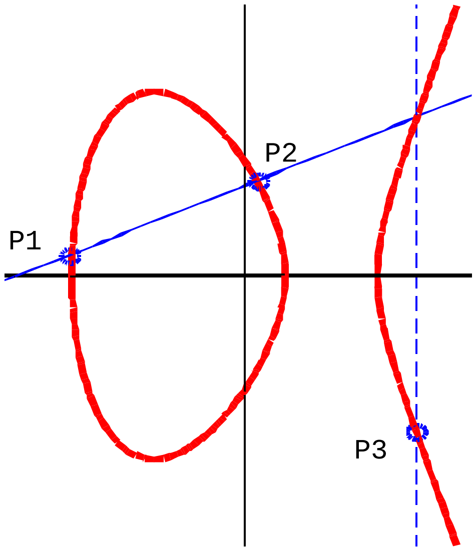


- Notice the *Points* in $\mathbb{F}_{23}$ graph

EC Point Addition addition formula

- We define a binary operation over E which makes E an Abelian group (Point Addition)
- Suppose:

$$\begin{aligned} &P, Q \in E \text{ and } P1 = (x_1, y_1), P2 = (x_2, y_2) \text{ and } P3 = (x_3, y_3) \\ &\text{we define} \quad P3 = P1 + P2 \end{aligned} \tag{2}$$



EC Point Addition addition formula

Suppose $P, Q \in E$ and $P1 = (x_1, y_1)$ and $P2 = (x_2, y_2)$ we define $P3 = P1 + P2$ and $P3 = (x_3, y_3)$

- EC Group Low in $\mathbb{F}(2^m)$, $P3 = P1 + P2$
- if $P1 \neq P2$ (ADD)

$$\begin{cases} \lambda = \frac{y_1+y_2}{x_1+x_2} \\ x_3 = \lambda^2 + \lambda + x_1 + x_2 + a_2 \\ y_3 = (x_1 + x_3)\lambda + x_3 + y_1 \end{cases}$$

- if $P1 = P2$ (DBL)

$$\begin{cases} \lambda = \frac{y_1}{x_1} + x_1 \\ x_3 = \lambda^2 + \lambda + a_2 \\ y_3 = (x_1 + x_3)\lambda + x_3 + y_1 \end{cases}$$

- For technical reason, we add a *point at infinity* to the *elliptic curve*, called $\mathcal{O}$
- Curves over different Fields $(\mathbb{R}, \mathbb{F}_{p^n}, \dots)$ can have different addition formula.

Elliptic Curve Discrete Logarithm Problem ECDSL

Basic operation in ECC is *point addition*(One-Way function)

- $kP = \underbrace{P + P + P + \cdots + P}_{k \quad times}$
- Suppose $Q = kP$ for some k. Given P and Q find k
- kP looks like $k - 1$ additions, But ...That's all elliptic curve cryptography implementation is about
- No efficient algorithm is known at this time to solve the ECDLP

Elliptic Curve cryptosystem implementation Options, (Top-Down)

1. Defining Equation for Elliptic curve
2. Representation of points
   - Affine Coordinates
   - Projective/Mixed Coordinates
   - ...
3. Scalar Multiplication technique kP ie. $kP = \underbrace{P + P + P + \cdots + P}_{k \quad times}$
   - Comb method
   - Window method
   - Scalar Recording
   - Parallel Processing
   - Security against side channel attack
4. Field Representation
   - Polynomial Basis

- Normal Basis
- Dual Basis

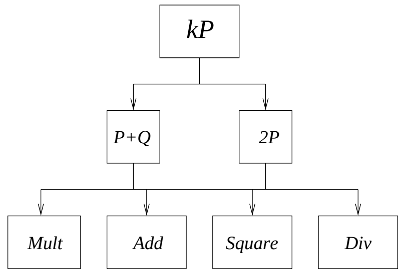
5. Finite Field operation Algorithm
   - Multiplication
   - Squaring
   - Inversion

- Speed of a ECC system is determined by the above factors.
- kP is the heart of an ECC, a fast kP means a fast cryptosystem

Elliptic Curve calculation, Arithmetic Hierarchy

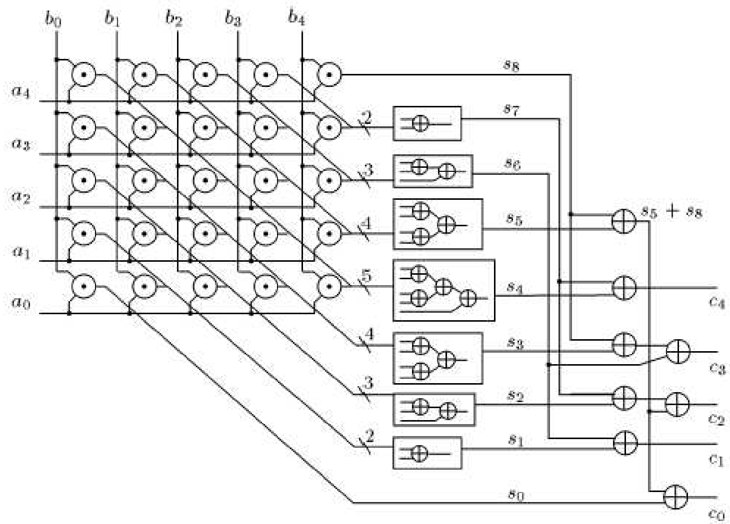
Scalar Multiplication Algorithm

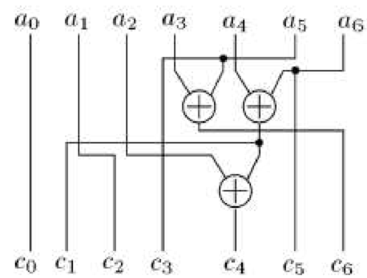
Finite Field Arithmetic



a_0	a_1	a_2	a_3	a_4	a_5	a_6

A member of GF(2^7)





- Multiplication in $GF(2^5)$ and Squaring in $GF(2^7)$

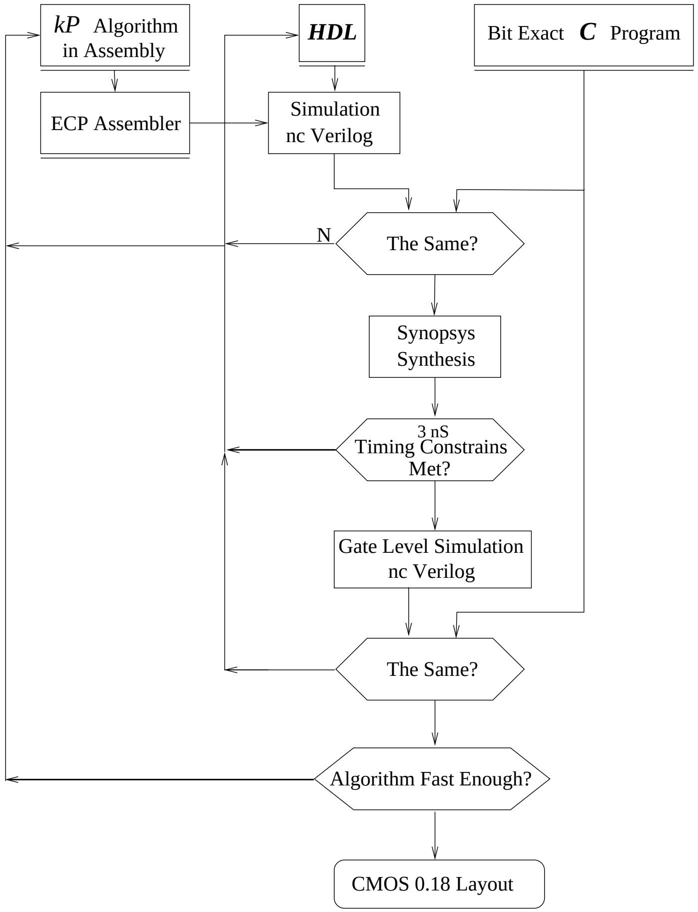
from: H. Wu, Bit-Parallel Finite Field Multiplier and Squarer. IEEE transaction on Computer. July 2002

Processor Features

The processor implements the following features to achieve high execution speed.

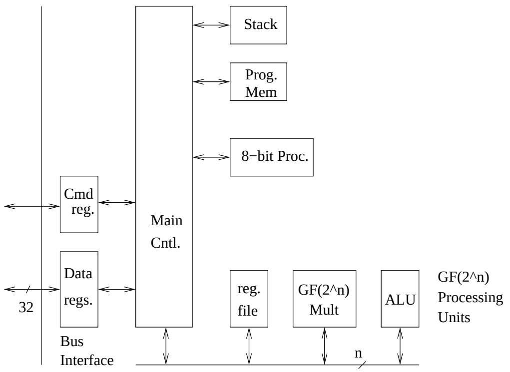
- VLIW architecture
- RISC type instruction set
- One cycle instruction execution
- Pipeline finite field multiplier

Design Flow

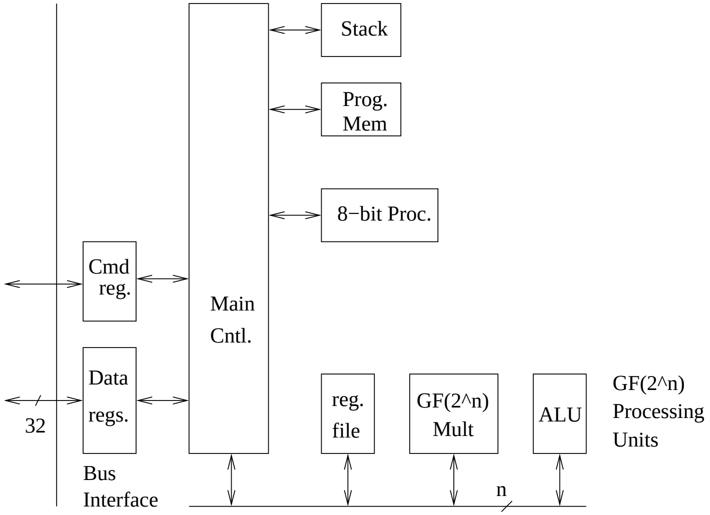

kP Algorithm in Assembly
HDL
Bit Exact ***C*** Program
ECP Assembler
Simulation nc Verilog
N
The Same?
Synopsys Synthesis
3 nS Timing Constrains Met?
Gate Level Simulation nc Verilog
The Same?
Algorithm Fast Enough?
CMOS 0.18 Layout

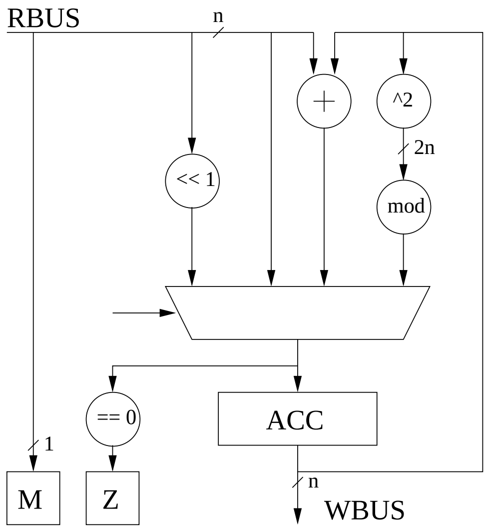
Architecture, General View

- The Finite Field Arithmetic unit is modeled on VLIW type architecture which permits parallel execution.
- The Finite Field processing unit consists of an ALU a Multiplier and a register file.
- Very small 8-bit processor is provided
- The communication with the host processor is implemented through utilization of a command register and a a data register.
- These units are controlled by the main control unit.



Architecture, General View





Finite Field multiplier

- Word Serial pipe line multiplier
- The size of the word is $m/4$ in $GF(2^m)$
- Architecture (HDL) is not hardwired to the size of the Galois Field.

Instruction Set

- The instructions set is sub divided into three categories.
  - Finite Field Arithmetic
  - Integer processing
  - Control Transfer
- The Compiler/Assembler analyzes the scalar multiplication program and detects Finite Field operations to be executed in parallel
- Such operations are packed into one "Finite Field Arithmetic" type instruction.

Instruction Set

8-bit processor	
`MOV rx, d8`	move immediate data to rx register
`DJNZ rx, addr`	decrement rx jump to addr if not zero
`DEC rx`, `INC rx`	increment rx, decrement rx
`SHL {c,rx}`, `SHL {rx,c}`	shift left Carry and rx
`MOV ry, rx`	move rx to ry
`REP`	repeat the next instruction r0 times
FF Arithmetic Unit	
`SQR A`	
`ADD A, Rx`	
`SHL A`	
`START Mul`	
`STOP Mul`	
`MOV Rx, P`	move product to Rx
`MOV Rx, A`	
`MOV A, Rx`	
`MOV S, Rx`	load multiplier register with Rx
Control Transfer	
`JMP flg,set, addr`	flg is Z (Zero flag), C (Carry flag), M (User flag)
`CALL flg,set, addr`	
`CLR M ,Set M`	
`HALT`	

Synthesis Result

The Processor is synthesized with Synopsys for $GF(2^{233})$

Unit	Area
Multiplier	1272102
ALU	28585
Squarer	4976
Register File	202799
Proc8	5617
Total	$\approx$ 1555271

- Critical Path 3.3 nSec $= T_A + [\log(m + m/4) - 1]T_X \approx 8T_X \approx 8 \times 0.3$
- Critical path is in the Parallel Multiplier

Performance

When implemented on a Xilinx Virtex 2000 FPGA: The processor can perform $10,000$ scalar multiplication per second on $GF(2^{167})$. Which is faster that the resent FPGA implementations.

Table 1: Performance of the Elliptic Curve Processor

Design	kP (mSec)	Inv. (Cycle)	$GF(2^m)$	FPGA (LUT, FF)	Clk (MHz)	FPGA	Year	
[2]	0.21		167	3000, 1769	76.6	XCV400E	2000	
[3]	0.143	326$=2m$	163	20068, 6321	66.4	XCV2000	2002	Supports Unnamed Curves
[5]	0.121+Inv.	>135 ?	233	19440, 16970	100	XCV6000	2003	
[4]	0.233	250	163	10017, 1930	66	XCV2000	2003	
Proposed	0.10	285	167	7562, 2364	66	XCV2000	2004	
Proposed	0.14	451	233	13900, 3164	66	XCV2000	2004	

References

[1] J. Lopez and R. Dahab "Fast Multiplication on Elliptic Curves over $GF(2^n)$ without Precomputation" *CHESS'99, Springer-Verlag, LNCS 1717, pp. 316-327* 1999

[2] Gerardo Orlando and Christof Paar "A High-Performance Reconfigurable Elliptic Curve Processor for $GF(2^m)$" *CHES 2000, LNCS 1965, pp. 41-56* 2000

[3] Nils Gura, Sheueling Chang Shantz, Hans Eberle, Sumit Gupta, Vipul Gupta, Daniel Finchelstein, Edouard Goupy, Douglas Stebila "An End-to-End Systems Approach to Elliptic Curve Cryptography" *Sun Microsystems Laboratories* 2002-2003

[4] Jonathan Lutz "High performance elliptic curve cryptographic co-processor" *Masters thesis, University of Waterloo* 2003

[5] C. Grabbe, M. Bednara, J. von zur Gathen, J. Shokrollahi, J. Teich "A High Performance VLIW Processor for Finite Field Arithmetic" *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'03)* 2003