On the Implementation of a Lightweight Generic FPGA ECC Crypto-Core over $GF(p)$

Martin Schramm
University of Applied Sciences
Deggendorf
Deggendorf, Germany
martin.schramm@hdu-deggendorf.de

Andreas Grzemba
University of Applied Sciences
Deggendorf
Deggendorf, Germany
andreas.grzemba@hdu-deggendorf.de

***Abstract*—State-of-the-art implementations of modern embedded computing platforms nowadays often feature reconfigurable components and/or programmable cores. Since these systems are more often connected to a larger network well-thought-out security mechanisms have to be put in place. In order to protect the assets of a (networked) system several security objectives have to be met. By including support for cryptographic operations inside the anyway available programmable chips it is suggested that the overall security level of the system can be raised. In this paper a concept of a lightweight generic FPGA core for performing elliptic curve cryptography (ECC) is proposed which underlying arithmetic calculations are solely conducted by a minimalistic arithmetic logic unit based on a single adder/subtracter instance. Due to the utilization of fairly plain algorithms it is well suited for the academic sector and can be seen as a quick guide for implementing cryptographic algorithms on reconfigurable hardware or also as a reference architecture for performing side-channel attacks such as simple/differential power analysis attacks. The main focus of this ongoing research work is the conceptual design, development and implementation of a reconfigurable FPGA-based hardware security module.**

I. Introduction

Embedded computing platforms have become a matter of course to be omnipresent in our everyday life. With the overall goal of making our life more comfortable, unfortunately also many concerns regarding system security arise which must not be underestimated. As stated in [1] reconfigurable hardware components, such as FPGAs, which are already included in many embedded designs can offer promising security features if additional security functionalities such as cryptographic cores are implemented correctly and the FPGA can act as a flexible hardware-based Trust Anchor. Elliptic curve cryptography schemes are preferred for the application of public-key schemes in embedded systems since for the same level of security the required key length can be held smaller compared to other pubic-key algorithms such as RSA [2] or ElGamal [3].This paper proposes a lightweight generic elliptic curve cryptographic core over prime fields.

The efficient implementation of finite field arithmetic is one of the most important prerequisites in elliptic curve systems due to the fact that curve operations are performed using these arithmetic operations of the underlying field. The order of a finite field is equal to the number of elements in the field. There exists a Galois Field of order q if and only if q is a prime power ($q = p^m$), where p is a prime number, and m is a positive integer. Within the scope of this paper only the case $m = 1$ and the resulting prime field $GF(p)$ is considered. An elliptic curve E over $GF(p)$ is defined by an equation of the form

$$E : y^2 \equiv x^3 + a \cdot x + b \, mod \, p, \tag{1}$$

where $a, b \in GF(p)$ satisfy $4 \cdot a^3 + 27 \cdot b^2 \not\equiv 0 \, mod \, p$. A pair $(x, y) \in GF(p)$, is a point on the curve if the coordinates satisfy equation (1). The point at infinity, denoted as $\mathcal{O}$, is also said to be on the curve.

The rest of the paper is structured as follows: Section II states the individual components the proposed elliptic curve crypto-core is based on. Section III lists the algorithms and equations necessary for performing modular arithmetic, and prime field arithmetic used by the ECC core. Section IV gives some possible application fields for this small and simple crypto-core and section V finalizes the paper with a short conclusion and a glance at the future work of the ongoing research work.

II. Implementation Details

This section lists the individual components the proposed generic ECC FPGA crypto-core is based on.

A. Elliptic Curve Crypto-Core

Figure 1 illustrates the high-level overview of the proposed ECC FPGA core. An Arithmetic Logic Unit (ALU) is responsible for performing simple arithmetic operations, such as simple addition and subtraction operations. A finite state machine within the ALU component is in charge of instructing the ALU to perform modular arithmetic operations, such as modular addition, modular multiplication, and the calculation of the multiplicative inverse in the prime field GF(p). The ALU component sends its status signals to the Arithmetic Unit Controller (AUC). On demand, the AUC finite state machine can send control signals to the ALU state machine in order to perform the elliptic curve operations point addition and point doubling. Finally the Main Controller (MC) finite state machine exploits the functionality of the AUC for the calculation of point multiplications, one of the most important arithmetic operations of elliptic curve cryptography.

Authorized licensed use limited to: SHANDONG UNIVERSITY. Downloaded on November 30,2021 at 02:03:37 UTC from IEEE Xplore. Restrictions apply.

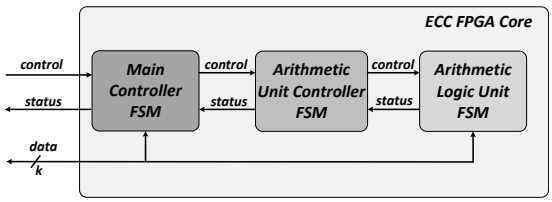

Fig. 1. High-level overview of ECC core

B. Arithmetic Logic Unit

The ALU of the proposed ECC crypto-core is intentional kept as small as possible due to two reasons. First of all the core is designed as a reference architecture for students which are investigating the possibility of performing side-channel attacks on reconfigurable hardware. Secondly the design is intended to be implemented in addition to existing FPGA configurations and needs to require as little space as possible. The internals are shown in figure 2.

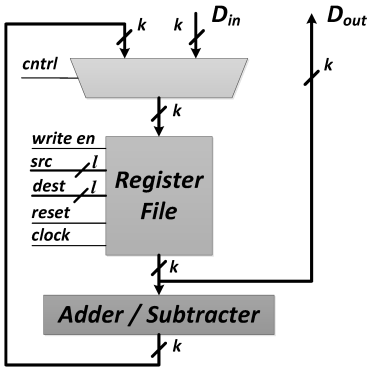

Fig. 2. Arithmetic Logic Unit

New k-bit data can be written to the register file of the ALU into a location denoted by the l-bit destination lines at falling clock edge if the write enable signal line is high. A multiplexer controls whether external data, or the result of the current ALU calculation will be written. Depending on the l-bit source signal lines data can be fed into the adder/subtracter instance preparing a new ALU operation or can be forwarded as ALU output.

C. Adder/Subtracter Instance

Within the proposed lightweight ECC core the adder/subtracter instance is the central component with which every arithmetic operation is performed (see figure 3). In detail it comprises different types of adders. In order to be able to calculate many consecutive additions in a row a k-bit carry save adder is used which basically consists of multiple parallel full adder (FA) cells. A carry save adder calculates two values out of three keeping the sum and the carry separated which keeps the latency low and the maximum possible frequency high. For the reassembling of the final result a k-bit carry select adder with a generic uniform m-bit block size is used. The use of a carry select adder is preferred as it offers a higher performance compared to a simple carry propagate adder. The number of blocks has a direct influence on the maximum achievable frequency. Subtractions are performed through additions using two's complement representation.

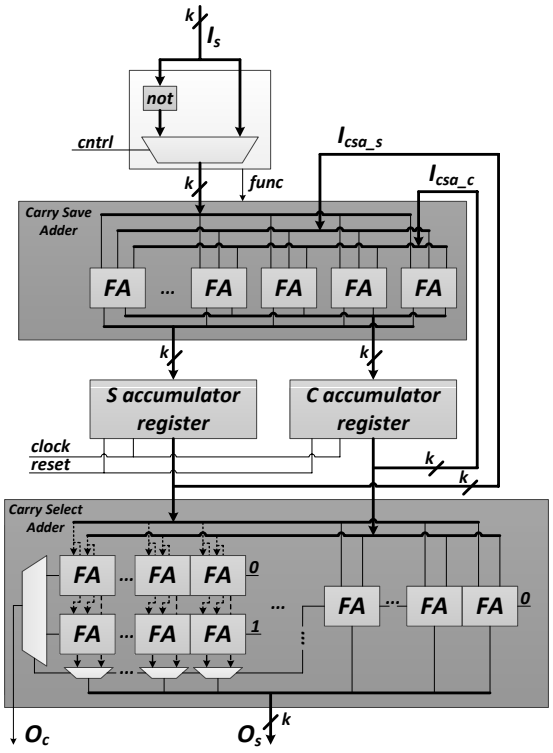

Fig. 3. Adder/Subtracter Instance

III. Utilized Algorithms

As it becomes apparent from figure 4 the operations in elliptic curve cryptography schemes are based on a hierarchical structure. ECC protocols make extensive use of an operation called point multiplication which on its part is based on point additions and point doublings. The mathematical foundation is given by the arithmetic of the chosen field, here the prime field GF(p). Prime field arithmetic consists of modular additions, modular multiplications and the calculation of the multiplicative inverse. Standard arithmetic operations such as simple additions and multiplications form the foundation of this hierarchy.

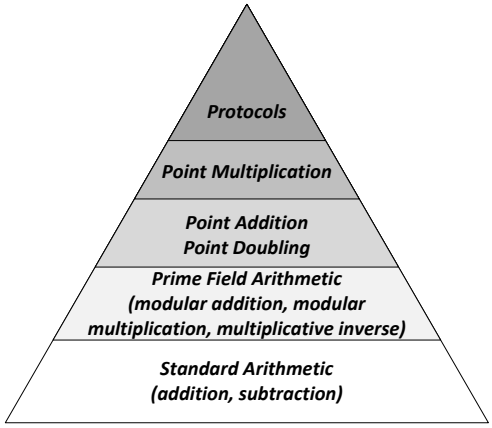

Fig. 4. Operation hierarchy in ECC schemes

Authorized licensed use limited to: SHANDONG UNIVERSITY. Downloaded on November 30,2021 at 02:03:37 UTC from IEEE Xplore. Restrictions apply.

Modular addition is the simplest of all prime field algorithms stated here. Modular addition ($S = A + B \, mod \, n$) only consists of a normal addition of the two input values A and B, which both are required to be smaller than the modulus n followed by a reduction step modulo n if necessary. If there is a carry out bit or the result is bigger than n the modulus will be subtracted from the intermediate result (see algorithm 1).

Alg. 1. Modular Addition

Input: $A \in [0, n-1], B \in [0, n-1], n$
Output: $S = A + B \, mod \, n$

$S <= A + B$;
if $carry = 1$ *or* $S \geq n$ **then**
  $S <= S - n$;
return S

Modular multiplication can also be achieved with nothing more than an adder/subtracter unit. Here, two k-bit values A and B are multiplied modulo n by a simple shift and add algorithm. Let b_i be the i-th bit of B (see algorithm 2). The result is calculated by scanning the k bits of B from left to right. If the i-th bit of B is a logic '1' than the intermediate result P is added to M modulo n. Independently of the bits of B each round the intermediate result P will be shifted left for one bit. If the most significant bit was '1' before the shift takes place, the modulus n will be subtracted from P. In other words this implements the modulo multiplication $2 \cdot P \, mod \, n$.

Alg. 2. Modular Multiplication

Input: $A \in [0, n-1], B \in [0, n-1], n$
Output: $M = A \cdot B \, mod \, n$

$P <= A$;
$M <= 0$;
for $i = 0$ ***to*** $k - 1$ **do**
  if $b_i = 1$ **then**
    $M <= (M + P) \, mod \, n$;
  $P <= 2 \cdot P \, mod \, n$;
return M

The multiplicative inverse of a number $A < p$, with p being a prime number can be calculated with the Extended Euclidean Algorithm (EEA) which computes the greatest common divisor together with the linear combination $gcd(A, B) = d = x \cdot A + y \cdot B$. For the proposed implementation a simplified version of the EEA which only outputs the multiplicative inverse is used. Let Y, B, D, X be auxiliary values and y_o, b_0, d_0, x_0 be the least significant bit of Y, B, D and X respectively. The multiplicative inverse A^{-1} of A is calculated as listed in algorithm 3.

Performing elliptic curve arithmetic operations requires point additions and point doublings. Point operations make use of the modular operations described above. In order to calculate the resulting point following equations are used, with s is computed differently corresponding to the needed operation (see equation (2) and (3) respectively).

Alg. 3. Multiplicative Inverse

Input: $A \in [0, p-1], p \; prime$
Output: $A^{-1} \, mod \, p$

$Y <= A, \; D <= p, \; B <= 1, \; X <= 0$;
while $Y \neq 0$ **do**
  while $y_0 = 0$ **do**
    $Y <= Y/2; \; B <= (B + b_0 \cdot p)/2$;
  while $d_0 = 0$ **do**
    $D <= D/2; \; X <= (X + x_0 \cdot p)/2$;
  if $Y \geq D$ **then**
    $Y <= Y - D; \; B <= (B - X) \, mod \, p$;
  else
    $D <= D - Y; \; X <= (X - B) \, mod \, p$;
return X

$$x_3 = s^2 - x_1 - x_2 \, mod \, p \tag{2}$$

$$y_3 = s \cdot (x_1 - x_3) - y_1 \, mod \, p \tag{3}$$

with:

$$s = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} \, mod \, p, & \text{(Point Addition)} \\ \dfrac{3 \cdot x_1^2 + a}{2 \cdot y} \, mod \, p, & \text{(Point Doubling)} \end{cases}$$

Elliptic curve point multiplication $Q = d \cdot P$ is the operation of successively adding the point P d-times along an elliptic curve. Algorithm 4 shows a simple way to perform point multiplication, the so-called double and add algorithm [4] quite similar to the square and multiply algorithm [5] for the fast calculation of modular exponentiation. Therefore the bits of d are scanned from left to right. If the i-th bit of d contains a logic '1' the point P will be added to the intermediate result by performing a point addition. In each round of the algorithm the intermediate result Q will be doubled by performing a point doubling operation. At the end of the algorithm the resulting point is $Q = d \cdot P$. Additionally a zero test is included in the hardware implementation to detect and report if the point at infinity $\mathcal{O}$ will be calculated.

Alg. 4. Point Multiplication

Input: Domain Parameters, P, d
Output: $Q = d \cdot P$

$Q <= \mathcal{O}$;
for i ***from*** m ***to*** 0 **do**
  $Q <= 2 \cdot Q$; (using point doubling)
  if $d_i = 1$ **then**
    $Q <= Q + P$; (using point addition)
return Q

IV. Possible application fields

This section briefly discusses some example applications in which the proposed lightweight ECC crypto-core could be employed.

Authorized licensed use limited to: SHANDONG UNIVERSITY. Downloaded on November 30,2021 at 02:03:37 UTC from IEEE Xplore. Restrictions apply.

A. Public Key Encryption

Using the described FPGA crypto-core for the realization of public-key encryption schemes confidentiality of messages and data can be provided. Examples are the Elliptic Curve Integrated Encryption Scheme (ECIES) [6] and the Provably Secure Elliptic Curve encryption scheme (PSEC) [7].

B. Signature Schemes

Furthermore, the proposed ECC core can also be used by signature schemes in order to provide data origin authentication, data integrity and non-repudiation. One example is the Elliptic Curve Digital Signature Algorithm (ECDSA) [8].

C. Key Establishment

Another possible application is the secure key agreement in order to provide two or more entities communicating over a potentially untrustworthy channel with a shared secret key. Chosen examples are the Elliptic Curve Diffie-Hellman (ECDH) key exchange protocol [9] as well as the elliptic curve analogue of the Station-To-Station (STS) protocol [10].

D. Reference Architecture

One very critical issue regarding the implementation of cryptographic algorithms on FPGA devices is the risk of side-channel attacks. Especially the double and add algorithm is prone to simple power analysis attacks [11]. In a power trace for a sequence of double and addition operations over a prime field the two operations can clearly be distinguished which potentially leads to the leakage of security sensitive information such as the secret key. The proposed design could act as a reference architecture for the investigation of side-channel attacks on reconfigurable hardware elements.

V. Conclusion and future work

This paper has reported on the implementation of a generic FPGA-based elliptic curve cryptography core utilizing an arithmetic logic unit deliberately kept simple. It has been shown that it is possible to implement a lightweight core with a simple adder/subtracter instance based on generic carry save adder and carry select adder cells on which standard arithmetic and prime field arithmetic operations can be executed. Finite state machines instruct the ALU to perform the steps required to calculate point additions, point doublings and finally point multiplications. The core was tested with the domain parameters for the five NIST-recommended randomly chosen elliptic curves over prime fields $GF(p)$ [12]. The primes of these sample parameters have a bit-length of $192, 224, 256, 384$ and 521, respectively. The core is well-suited for the demonstration of elliptic curve encryption and signature schemes as well as key agreements over prime fields. In addition it could be seen as a basis for the investigation of side-channel attacks and the implementation of side-channel attack resistant cores.

In the next steps of this ongoing research work the efficiency of the core should be further improved for instance by replacing the state machines by RISC controller. For the overall project goal the risk of side-channel attacks should be mitigated by adding dummy operations [13] and implementing time invariant algorithms [14] with which an attacker can get no information of timing and power consumption measurements. Furthermore the next steps also include the implementation of a generic ECC core over the binary field $GF(2^m)$ with m being a prime number. The operations in $GF(2^m)$ are typically easier to implement in hardware than their counterparts in prime fields $GF(p)$ because bitwise addition in $GF(2^m)$ does not have any carry propagation.

Acknowledgment

This research work has been supported by the research project no. 16BY1209B of the German Federal Ministry of Education and Research.

References

[1] Schramm, M., Grzemba, A., *Reconfigurable Trust for Embedded Computing Platforms*, IEEE Applied Electronics International Conference, 2012

[2] Ghoreishi, S. et al., *High Speed RSA Implementation Based on Modified Booth's Technique and Montgomery's Multiplication for FPGA Platform*, Proceedings of the 2009 Second International Conference on Advances in Circuits, Electronics and Micro-electronics, 2009

[3] Tawalbeh, L., Sweidan, S., *Hardware Design and Implementation of ElGamal Public-Key Cryptography Algorithm*, Information Security Journal: A Global Perspective, Volume 19 Issue 5, 2010

[4] Paar, C. and Pelzl, J, *Understanding Cryptography: A Textbook for Students and Practitioners*, Springer-Verlag New York, Inc., 2010

[5] Gudu, T., *A New Scalable Hardware Architecture for RSA Algorithm*, International Conference on Field Programmable Logic and Applications, 2007

[6] Gayoso M. et al., *A comparison of the standardized versions of ECIES*, Sixth International Conference on Information Assurance and Security, 2010

[7] Okamoto, T., Fujisaki, E., Morita H., *PSEC: Provably Secure Elliptic Curve Encryption Scheme*, IEEE P1363a, 2000

[8] FIPS PUB 186-3, *Digital Signature Standard (DSS)*, Federal Information Processing Standards Publication, 2009

[9] National Institute of Standards and Technology, *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography*, Special Publication 800-56A, 2006

[10] Hankerson, D., Menezes, A., Vanstone S., *Guide to Elliptic Curve Cryptography*, Springer-Verlag New York, Inc., 2003

[11] Li, H. et al., *Simple power analysis attacks using chosen message against ECC hardware implementations*, World Congress on Internet Security (WorldCIS), 2011

[12] National Institute of Standards and Technology, *Recommended Elliptic Curves for Federal Government Use*, Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004

[13] Moeller, B., *Securing Elliptic Curve Point Multiplication against Side-Channel Attacks*, Lecture Notes in Computer Science, 2001

[14] Byrne, A. et al., *SPA resistant Elliptic Curve Cryptosystem using Addition Chains*, Proceedings of the International Conference on Information Technology, 2007

Authorized licensed use limited to: SHANDONG UNIVERSITY. Downloaded on November 30,2021 at 02:03:37 UTC from IEEE Xplore. Restrictions apply.