A VHDL Generator for Elliptic Curve Cryptography

Kimmo Järvinen, Matti Tommiska, and Jorma Skyttä

Helsinki University of Technology
Signal Processing Laboratory
Otakaari 5 A, FIN-02150, Finland
{kimmo.jarvinen,matti.tommiska,jorma.skytta}@hut.fi

Abstract. A VHDL generator called SIG-ECPM is presented in this paper. SIG-ECPM generates fully synthesizable and portable VHDL code implementing an elliptic curve point multiplication, which is the basic operation of every elliptic curve cryptosystem. The use of automated design flow significantly shortens design times and reduces error proneness.

1 Introduction

Elliptic Curve Cryptography (ECC) has been of much interest in the world of cryptography during the past few years, because a high level of security can be achieved with considerably shorter keys than with conventional public-key cryptography, e.g. RSA [1]. Because of the short keys, ECC is an attractive alternative in applications, where bandwidths or memory resources are limited [2]. The basic operation of any elliptic curve cryptosystem is the Elliptic Curve Point Multiplication (ECPM) defined as $Q = kP$ where Q and P are distinct points on an elliptic curve E and k is a large integer [1].

Automation of the design flow significantly shortens design times and reduces error proneness of the process. A Hardware Description Language (HDL) generator called SIG-ECPM is presented in this paper. SIG is an acronym for the Signal Processing Laboratory at Helsinki University of Technology. SIG-ECPM generates fully synthesizable and device independent VHDL-code implementing ECPM, when a set of parameters is given. Use of SIG-ECPM is beneficial in applications where several designs have to be implemented in short time. The research work was performed in the GO-SEC project at HUT (see `go.cs.hut.fi`).

2 The VHDL Generator

There is a large variety of recommended curves which can be used. Because the design process of an ECPM implementation is complex, it would be too complicated to implement every design by hand. Thus, a VHDL generator, which automatically generates synthesizable VHDL-code implementing an ECPM, was designed. This generator is called SIG-ECPM and it was written in the C programming language.

J. Becker, M. Platzner, S. Vernalde (Eds.): FPL 2004, LNCS 3203, pp. 1098–1100, 2004.

ECPM architecture utilized by SIG-ECPM is presented in [3]. The architecture implements entire ECPM while traditionally only Galois field operations are implemented. It uses the point multiplication algorithm developed by López and Dahab in [4]. An inverter proposed by Shantz in [5] was used for Galois field inversions. Multiplier structure was optimized especially for the 4-to-1-bit LUT structure of FPGAs. Details of the multiplier architecture are also presented in [3]. To achieve maximum performance, the ECPM architecture was designed so that the Galois field cannot be changed in the design, i.e. non-reconfigurable designs are created [3]. Thus, a different design has to be designed for every Galois field, which justifies the use of a VHDL generator. The use of SIG-ECPM vastly reduces the disadvantages of non-reconfigurable implementations.

Parameters are given for SIG-ECPM and it generates VHDL-code, which implements ECPM on a curve described by the parameters. The parameters are given as a parameter file of about 25 lines of text. SIG-ECPM generates over 7000 lines of VHDL-code into 26 files. Thus, it is obvious that SIG-ECPM saves a considerable amount of design time. An example of a parameter file, implementing a curve called sect113r1 recommended by SECG (Standards for Efficient Cryptography Group) in [6], is presented in Fig. 1.

```
**** PARAMETER-FILE FOR SIG-ECPM ****
*** Curve Sect113r1

*** Field Specifications.
FIELD {
    m: 113;
    irrpolycoeffs: 3;
    irreducible: 113 9 0;
};

*** Elliptic Curve Specifications.
ECC {
    fixedP: 1;
    Px: 09D73616F35F4AB1407D73562C10F;
    Py: 0A52830277958EE84D1315ED31886;
    fixedC: 1;
    a: 03088250CA6E7C7FE649CE85820F7;
    b: 0E8BEE4D3E2260744188BE0E9C723;
    tvectors: 5;
};

*** Multiplier Specifications.
MULT {
    multipliers: 2;
    latency: 12;
    tvectors: 5;
};

END;
```

Fig. 1. A parameter file

Parameters of Fig. 1 consist of field, elliptic curve and multiplier specifications. The field specifications define the Galois field over which the elliptic curve E is defined. In Fig. 1, the field is $GF(2^{113})$ and the irreducible polynomial generating this field is $m(x) = x^{113} + x^9 + 1$. The elliptic curve specifications define the elliptic curve E and point P. These values are optional for SIG-ECPM, but fixing E and P results in slightly faster and smaller implementations. The number and structure of Galois field multiplier(s) is defined in the multiplier specifications. The use of a second multiplier speeds up ECPM calculation significantly [3]. Latency constraint sets an upper bound for the latency of a single Galois field multiplication, which is the critical operation of ECPM. SIG-ECPM optimizes multiplier so that it satisfies this constraint.

3 Results

Several ECPM implementations using elliptic curves recommended by the Standards for Efficient Cryptography Group (SECG) in [6] were created with SIG-ECPM. Very competitive results were obtained, e.g. a SIG-ECPM design calculates an ECPM on sect163r1 curve in 106 μs when Xilinx Virtex-II XC2V8000-5 FPGA was used as a target device. The design flow used for the designs is presented in Fig. 2 with typical run-times.

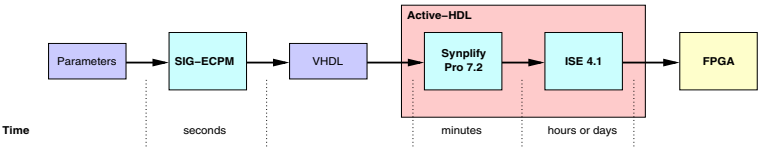

Fig. 2. Design flow with typical run-times for the designs created with SIG-ECPM

The benefits of SIG-ECPM are obvious when the time required for the design process is considered. It would require days to make the entire design by hand. With SIG-ECPM, the same work can be done in a couple of seconds. However, the synthesis and implementation (place & route) still require a considerable amount of time. The real bottleneck of the design process is the implementation, which requires time from hours to even days. Anyhow, design time can be significantly shortened with a VHDL-generator, such as SIG-ECPM.

4 Conclusions

A VHDL generator called SIG-ECPM was presented in the paper. SIG-ECPM generates fully synthesizable and portable VHDL code implementing an ECPM. The use of a VHDL generator speeds up the design process significantly and reduces error proneness. VHDL generators are very useful in ECC applications, where a large variety of different parameters (curves, Galois fields, etc.) are used. They reduce the disadvantages of fast non-reconfigurable ECPM implementations, thus, allowing faster performance.

References

1. Blake, I., Seroussi, G., Smart, N.: Elliptic Curves in Cryptography, London Mathematical Society Lecture Note Series 265. Cambridge University Press (2002)
2. Bednara, M., Daldrup, M., von zur Gathen, J., Shokrollahi, J., Teich, J.: Reconfigurable Implementation of Elliptic Curve Crypto Algorithms. Proceedings of the International Parallel and Distributed Processing Symposium, IPDPS'02, 9th Reconfigurable Architectures Workshop, RAW-2002, Marriott Marina, Fort Lauderdale, Florida, USA (2002) 157 – 164
3. J rvinen, K., Tommiska, M., Skytt , J.: A Scalable Architecture for Elliptic Curve Point Multiplication. Submitted to the International Conference on Field Programmable Technology, FPT 2004 (2004)
4. López, J., Dahab, R.: Fast Multiplication on Elliptic Curves over $GF(2^m)$ without Precomputation. Proceedings of Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, LNCS 1717, Springer-Verlag (1999) 316 – 327
5. Shantz, S.C.: From Euclid's GCD to Montgomery Multiplication to the Great Divide. Technical Report SMLI TR-2001-95, Sun Microsystems Laboratories (2001)
6. Certicom Research: SEC 2: Recommended Elliptic Curve Domain Parametres. Standards for Efficient Cryptography (2000) URL: `http://www.secg.org/collateral/sec2_final.pdf`, (visited: June 3, 2004).