An Efficient Scheme for Implementation of SM2 Digital Signature over GF(p)

Yanhua Liu, Wei Guo, Ya Tan, Jizeng Wei, and Dazhi Sun

School of Computer Science and Technology, Tianjin University, Tianjin 300072, China
{yanhanliu,weiguo,tanya,jizengwei,sundazhi}@tju.edu.cn

Abstract. In this paper, we proposed an efficient implementation scheme for digital signature based on the cryptography algorithm SM2, which is established as the Elliptic Curve Cryptography (ECC) standard of China. Algorithm analysis reveals speed bottleneck lies in scalar multiplication, which is time consuming for the master processor to implement. Therefore, a configurable ECC coprocessor is employed in the scheme to improve the processing speed. In order to improve the efficiency of data transport within digital signature, a fine-grained programming and high Instruction Level Parallelism architecture is employed. To decrease intermediate registers, point doubling algorithm is optimized to reduce space complexity. The speed of critical steps within SM2 digital signature is improved significantly by the coprocessor. With these improvements, scalar multiplication can be achieved in 3 ms at 80 MHz for 192-bit ECC. The results show that our scheme is competitive for embedded platforms.

Keywords: SM2 digital signature, ECC, point doubling, scalar multiplication, coprocessor.

1 Introduction

The rapid progress in e-commerce on mobile handheld services has introduced new challenges to researchers working on digital signature of embedded platforms. ECC algorithms are considered better candidates for digital signature in the resource constraint environment because they require shorter key sizes to achieve the same security strength. Elliptic Curve Digital Signature Algorithm (ECDSA) has been widely used in various applications. The State Cryptography Administration of China also promulgated the digital signature of cryptography algorithm SM2, [1] which is an alternative to ECDSA, in December 2010.

Scalar multiplication is the core operation of SM2 digital signature. Scalar multiplication contains point doubling and addition, which are composed of basic modular arithmetic operations, such as modular multiplication, addition, and subtraction. Two new architectures are proposed by Miaoqing, Kris and Tarek [2] to achieve the critical operation Montgomery modular multiplication. To decrease data delay, these architectures precomputed partial results using two possible assumptions of input data. In order to speed up the elliptic curve point calculation, point addition and doubling algorithms are optimized by Dimitrios et al [3] to fit hardware implementation. While algorithmic-level improvements are continually proposed, efficient mapping and scheduling between hardware and software is becoming more and more

V. Khachidze et al. (Eds.): iCETS 2012, CCIS 332, pp. 250–258, 2012.

important. Sergey Morozov proposed system integration on an OMAP Platform for ECC in 2011 [4], distributing underlying field operations on the DSP and minimizing data exchange between ARM and DSP. To accelerate scalar multiplication over the general prime field, a high speed coprocessor based upon the Residue Number System is proposed by Guillermin [5], guaranteeing carry-free arithmetic and easy parallelism. And Raveen R. Goundar [6] also presented co-Z addition formulae for scalar multiplication on Weierstraβ elliptic curves to reduce time complexity and space complexity. However, to our knowledge, there is almost no relevant research work of software/hardware cooperative implementation for SM2 digital signature. As cryptography algorithm SM2 is an elliptic curve cryptography standard in China, which will be widely used for e-commerce in the future, it is very promising to research software/hardware cooperative implementation.

In this paper, an efficient implementation for SM2 digital signature scheme is presented.

- Point doubling algorithm is optimized to achieve an area-efficient architecture.
- Since digital signature scheme requires large computational resources, which strongly restricts software implementation, a software/hardware cooperative System-on-Chip (SoC) implementation scheme for SM2 digital signature is proposed.
- A TTA-like configurable ECC coprocessor is designed to exploit Instruction Level Parallelism and reduce redundant data transport.

The rest of the paper is organized as follows: in section 2, a brief algorithm analysis of SM2 digital signature and optimized point doubling algorithm are presented. An efficient scheme of SM2 digital signature for embedded platform is proposed in section 3. We then give performance evaluations in section 4. Finally, the conclusion of this work is summarized in section 5.

2 Algorithm Analysis and Optimization

2.1 SM2 Digital Signature

The digital signature of cryptography algorithm SM2 is based on ECC. The security of the protocol depends on the intractability of the Elliptic Curve Discrete Logarithm Problem (ECDLP). Meanwhile, the widely used ECDSA is also based on ECDLP. Acting as ECC standard of China, SM2 digital signature is an alternative of ECDSA, which will play an important role in e-commerce. A brief outline of SM2 digital signature is given below [1].

Assuming that user A signs the message M and generates signature according to the following steps of SM2 digital signature generation:

After receiving the signature, user B verifies signature (r, s) on the message M with public key P_A according to the following steps of SM2 signature verification:

2.2 An Optimized Algorithm of Point Doubling

Scalar multiplication dominates the performance of digital signature, and point doubling is the key component of scalar multiplication. In this section, an optimized point doubling algorithm is proposed.

Point doubling is programmable as a sequence of modular operations (multiplication, inversion, addition, and subtraction). Among these operations, modular inversion is most time consuming. A fast algorithm for point doubling is described in [8]. The algorithm adopts Jacobian coordinates to avoid the modular inversion in *Fp*. Based on this algorithm; we present a new alternative algorithm which is area efficient for hardware implementation. The algorithm is presented below:

Proposed Algorithm. Point doubling ($y^2 = x^3 - 3x + b$)

Input: Elliptic curve $y^2 = x^3 - 3x + b$, the point on the curve $P = (X_1, Y_1, Z_1) \in Fp$

Output: $Q = 2P = (X_3, Y_3, Z_3)$

1.	$U = Z_1^2$	$(U = Z_1^2)$
2.	$V = U^2$	$(V = Z_1^4)$
3.	$U = V + V$	$(U = 2Z_1^4)$
4.	$U = U + V$	$(U = 3Z_1^4)$
5.	$V = X_1^2$	$(V = X_1^2)$
6.	$W = V + V$	$(W = 2X_1^2)$
7.	$W = V + W$	$(W = 3X_1^2)$
8.	$V = U - V$	$(V = 3Z_1^4 - 3X_1^2)$
9.	$Y_3 = Y_1 + Y_1$	$(Y_3 = 2Y_1)$
10.	$Z_3 = Y_3 Z_1$	$(Z_3 = 2Y_1 Z_1)$
11.	$Y_3 = Y_3^2$	$(Y_3 = 4Y_1^2)$
12.	$W = Y_3 X_1$	$(W = 4X_1 Y_1^2)$
13.	$U = Y_1^2$	$(U = Y_1^2)$
14.	$Y_3 = Y_3 U$	$(Y_3 = 4Y_1^4)$
15.	$Y_3 = Y_3 + Y_3$	$(Y_3 = 8Y_1^4)$
16.	$X_3 = V^2$	$(X_3 = (3Z_1^4 - 3X_1^2)^2)$
17.	$U = W + W$	$(U = 8X_1 Y_1^2)$
18.	$X_3 = X_3 - U$	$(X_3 = (3Z_1^4 - 3X_1^2)^2 - 8X_1 Y_1^2)$
19.	$U = W - X_3$	$(U = 4X_1 Y_1^2 - X_3)$
20.	$U = VU$	$(U = (3Z_1^4 - 3X_1^2)(4X_1 Y_1^2 - X_3))$
21.	$Y_3 = U - Y_3$	$(Y_3 = (3Z_1^4 - 3X_1^2)(4X_1 Y_1^2 - X_3) - 8Y_1^4)$

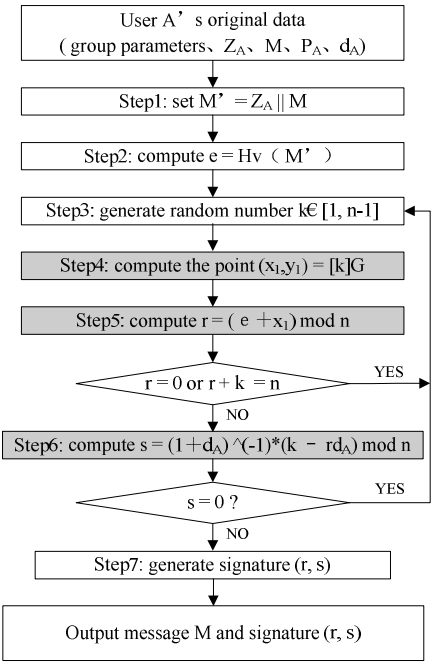
Here I, M, and S are the cost of inversion, multiplication, and squaring, respectively. Typically, I $\approx$ 100M, M $\approx$ 0.8S [6]. The computational complexity of this algorithm is 4M+6S, as is that of Li et al [8]. However, the number of required intermediate variables is less than the number presented by Li et al [8]. So that space complexity is lower, which is critical for resource constraint embedded platform, and

due to the reality that the cost of addition is negligible compared with multiplication, we apply addition in step3 and step4 of proposed algorithm instead of multiplication U=3*V. Furthermore, as the function units of modular multiplication can achieve modular squaring efficiently enough, a squarer circuit is omitted for area reduction reasons.

3 Implementation Scheme

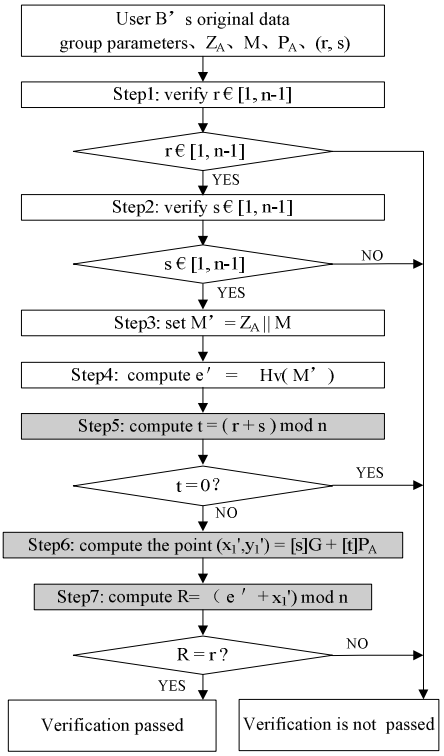
Since SM2 digital signature requires large computational resources, which strongly restricts software implementation, hardware design can assist in achieving the critical steps within digital signature efficiently. A software/hardware cooperative SoC implementation scheme is proposed in this section.

To speed up scalar multiplication, a cryptographic processor is designed. Nevertheless, SM3 cryptographic hash algorithm contains a large amount of iterative and compressing operations. It is better to be implemented by master processor, to achieve a satisfactory balance between speed and area. Master processor is not good at modular reduction because a large amount of time is required for consuming mod operations, so we employed the coprocessor to overcome the limited processing capacity of master processor. As a result, in Fig.1 and Fig.2, the steps in black are implemented by the coprocessor, and the others by master processor on the SoC implementation platform.



- Z_A denotes the ID information of user A
- M denotes the message
- P_A denotes the public key
- d_A denotes the secret key
- M' = Z_A || M :generation a message which connects ID information Z_A with message M
- e denotes the hash value of message M'
- Hv denotes the hash function
- G denotes the base point
- [k]G denotes the scalar multiplication with scalar k and base point G
- r denotes part of SM2 digital signature
- s denotes the rest part of SM2 digital signature

Fig. 1. Flow of SM2 signature generation



- Z_A denotes the ID information of user A
- M denotes the message
- P_A denotes the public key
- (r , s) denotes the digital signature which user B has received .
- M' = Z_A || M :generation a message which connects ID information Z_A with message M
- e' denotes the hash value of message M'
- Hv denotes the hash function
- G denotes the base point
- [s]G denotes the scalar multiplication with scalar s and base point G
- [t]P_A denotes the scalar multiplication with scalar t and point P_A (public key)
- [s]G+[t]P_A means calculate point [s]G and [t]P_A, then add the two point.

Fig. 2. Flow of SM2 signature verification

3.1 Implementation Platform

The block diagram of a SoC implementation platform for evaluating the proposed scheme is shown in Fig.3, which is implemented on Xilinx V5 FPGA. MicroBlaze CPU acts as master processor, within which a random number is generated and SM3 cryptographic hash algorithm is accomplished. In the meantime, it schedules the data stream in the evaluation platform. ECC coprocessor accomplishes scalar multiplication, which affects the overall performance. Meanwhile, it also takes charge of basic modular arithmetic operations and fast transmission for huge amounts of data is provided by DMA. TIMER and UART operate together for displaying the intermediate data to verify correctness and determine the speed bottleneck.

3.2 The Proposed ECC Cryptographic Processor

To speed up large operand operations, an ECC coprocessor based on Transport Triggered Architecture (TTA) for hardware implementation of scalar multiplication is proposed. TTA is a configurable architecture for special application, which is similar to VLIW to achieve Instruction Level Parallelism for programming. The most important feature of TTA is programming fine-grained data transports, which adds an extra level of control to the code generation.

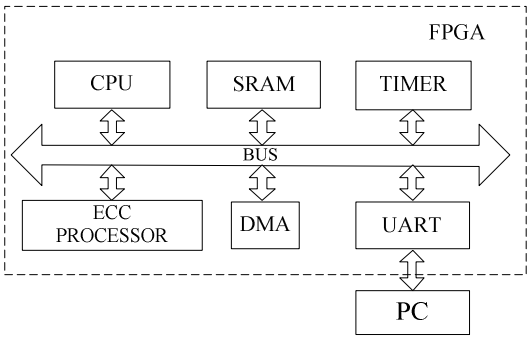


Fig. 3. Implementation platform for SM2 digital signature

As illustrated in Fig.4, this ECC cryptographic processor consists of three modules, including program flow control unit, function units and transport network. The processor's FUs are composed of Jump Unit (JMP), Arithmetic Logic Unit (ALU), Load and Store Unit (LDST), Large operand Addition/Subtraction Unit (LADSB), Multiplier Unit (LMUL) and Montgomery Multiplier Accumulators Unit (MMAU). The units of JMP and ALU implement operation jump, shifting and comparing. Data load and store is accomplished by LDST units. The LADSB unit supports addition and subtraction for large operand whose length more than bus width, while the LMUL and MMAU unit are used together to implement the FIOS Montgomery modular multiplication. The operation of outer loop in FIOS Montgomery modular multiplication is achieved by LMUL units and the inner loop operation by MMAU.

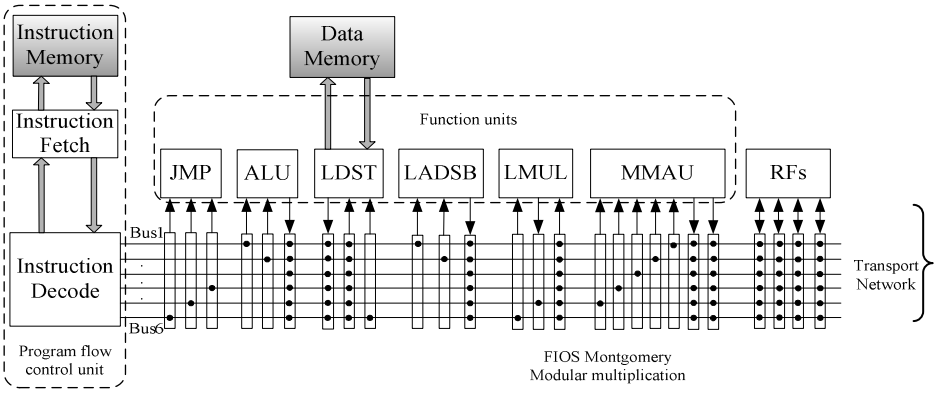


Fig. 4. A TTA-like ECC cryptographic processor

4 Implementation Results and Analysis

We have mapped our implementation platform for SM2 digital signature onto a Xilinx Virtex-5 XC5V-LX110T FPGA to obtain quantitative performance, for comparison

purposes. The ECC coprocessor implementation can run at 80 MHz, occupies 13664 LUTs, and uses 14 DSP48E blocks as well as 16K RAM. FPGA design is highly adaptable and easily reprogrammable for varying prime field sizes, which makes it suitable for ECC, due to the large number of different secure curves, prime fields currently available. As scalar multiplication is time consuming for the master processor to implement, so an ECC coprocessor is design on FPGA to improve processing speed. Compared with other work, this work can achieve higher clock frequency and take less delay with smaller area cost. The result of the comparison is listed in Table 1. Our clock frequency is relatively faster, because a two-phase hybrid pipelining architecture, which includes transport pipelining and FU pipelining, is adopted to execute instruction. Sakiyama et al [9] propose a reconfigurable data path for RSA and ECC, but a 260 bit width data path is selected for 256-bit ECC, while the modular arithmetic units presented by McIvor, McLoone and McCanny [10] adopted 256-bit for large number processing. Kendall, Hamad and Daler [11] choose the large bit width for data path because routing signals requires many stages of generic bit-level FPGA switches. However, 32 bit data path is chosen in this work, which is another factor attributing to fast clock frequency. Since modular multiplication applied in the scheme is a key operation, it affects the overall performance of scalar multiplication. Sakiyama et al [9] adopted d-digit serial Montgomery modular multiplication and full word Montgomery modular multiplication is applied by McIvor et al [10], while Kendall et al [11] employed a regular (non-modular) multiplier coupled with modular reduction. Compared with these algorithms, FIOS Montgomery modular multiplication adopted in this work can offer less delay. So that execution time of scalar multiplication in this work is less compared with those designs [9], [10], and [11]. The area of the proposed design is relatively smaller, due to the fact that large operands of ECC can be efficiently converted into the 32-bit word length to operate in modular arithmetic units. While large operand is operated directly within modular arithmetic units in the works of Sakiyama et al and McIvor et al, [9-10] the area cost of Kendall et al.'s design [11] is increased to support the max key size 521 bit. Thus, taking these facts into consideration, the proposed coprocessor outperforms each of these designs [9], [10], [11].

Table 1. The comparison of different scalar multiplication implementations

Reference	Target Platform	Key size	Area	Performance [msec]	Max Clock Freq[MHZ]
[9]	XC3S5000	256	27,597 slices	17.7	40
[10]	XC2VP125	256	15,755 slices	3.86	40
[11]	XC2VP100-6	521	20,793 slices	7.24	50
This work	XC5V-LX110T	192	3,416 slices	3.0	80

To our knowledge, there is little relevant work about software/hardware cooperative implementation of SM2 digital signature. However, an efficient implementation of SM2 digital signature, which acts as the digital signature standard for e-commerce in China, is significant. Our scheme is implemented with the Micro Blaze as master processor and a TTA-like ECC coprocessor at 75 MHz. Peak performance of 311 times per second scalar multiplication for 192-bit ECC has been achieved, and the time needed for other critical steps is shown in Table 2. As SM3 cryptographic hash algorithm contains a large amount of iterative and compressing operations, it is better to implement it in the master processor to achieve a satisfactory balance between speed and area. Moreover, software implementation of hash algorithm costs only 0.128s. From Table 2, it can be seen that the main operation scalar multiplication is accelerated by coprocessor efficiently. Compared to software implementation scalar multiplication in the work of Haodong and Qun, [12] which costs 1.60s with an 8 MHz, 16-bit CPU for 160-bit ECC, the time of hardware implementation in this work just is 3.221 ms for 192-bit ECC. Modular reduction and inversion must be applied within SM2 digital signature during step 6 in the signature generation process. These are also performance critical operations. By taking advantage of ECC coprocessor, this operation can be achieved within 0.227 ms without any area cost, and the speed is significantly improved. This result shows that the scheme is an efficient solution to meet the requirement of SM2 digital signature on embedded platform.

Table 2. The performance of critical steps for SM2 digital signature

Function	Critical steps	Platform	Performance
Signature generation	Step2: $e = Hv(M')$	MicroBlaze	0.128s
	Step4: $(x_1, y_1) = [K]G$	ECC coprocessor	3.221ms
	Step5: $r = (e + x_1) \bmod n$	ECC coprocessor	0.173us
	Step6: $s = (1 + d_A)^{-1}(k - rd_A)$	ECC coprocessor	0.227ms
Signature verification	Step4: $e = Hv(M')$	MicroBlaze	0.128s
	Step5: $t = (r + s) \bmod n$	ECC coprocessor	0.106us
	Step6: $(x_1, y_1) = [s]G + [t]P_A$	ECC coprocessor	6.464ms
	Step7: $R = (e + x_1) \bmod n$	ECC coprocessor	0.173us

5 Conclusion

In this paper, an efficient software/hardware cooperative SoC implementation for SM2 digital signature scheme is presented. An optimized point doubling algorithm is applied to decrease hardware cost. A TTA-like ECC coprocessor is designed to accelerate key operation scalar multiplication. In comparison to other works, the ECC coprocessor can obtain a satisfactory balance between speed and area for scalar multiplication. Herewith, SM2 digital signature speeds up significantly, and the core

operation scalar multiplication can be accomplished in 3 ms at 80 MHz with 3416 slices on Xilinx Virtex-5 FPGA. The results show that the proposed SM2 digital signature implementation meets the requirements of the embedded platform efficiently.

Acknowledgement. This work is supported in part by both the Natural Science Foundation of Tianjin (No. 11JCZDJC15800) and the National Natural Science Foundation of China (No. 61003306).

References

1. State Cryptography Administration of China: Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves, http://www.oscca.gov.cn
2. Miaoqing, H., Kris, G., Tarek, E.G.: New Hardware Architectures for Montgomery Modular Multiplication Algorithm. IEEE Transactions on Computers 60(7) (2011)
3. Dimitrios, M.S., Apostolos, P.F., Harris, E.M., Athanasios, P.K., Thanos, S.: An RNS Implementation of an Fp Elliptic Curve Point Multiplier. IEEE Transactions on Circuits and Systems—I: Regular Papers 56(6), 1202–1213 (2009)
4. Sergey, M.: System Integration of Elliptic Curve Cryptography on an OMAP Platform. In: 2011 IEEE 9th Symposium Application Specific Processors (SASP), pp. 52–57. IEEE Computer Society, Los Alamitos (2011)
5. Guillermin, N.: A High Speed Coprocessor for Elliptic Curve Scalar Multiplications over Fp. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 48–64. Springer, Heidelberg (2010)
6. Raveen, R.G., Marc, J., Atsuko, M., Matthieu, R., Alexandre, V.: Scalar Multiplication on Weierstrass Elliptic Curves from Co-Z Arithmetic. Journal of Cryptographic Engineering 1, 161–176 (2011)
7. State Cryptography Administration of China: SM3 Cryptographic Hash Algorithm, http://www.oscca.gov.cn
8. Li, M., Wu, D., Dai, K., Zou, X.C.: Research and Design of a High-Performance Scalable Public-Key Cipher Coprocessor. ACI'A Ele Ironica Sinica 3(3), 665–669 (2011)
9. Sakiyama, K., Mentens, N., Batina, L., Preneel, B., Verbauwhede, I.: Reconfigurable Modular Arithmetic Logic Unit for High-Performance Public-Key Cryptosystems. In: Bertels, K., Cardoso, J.M.P., Vassiliadis, S. (eds.) ARC 2006. LNCS, vol. 3985, pp. 347–357. Springer, Heidelberg (2006)
10. McIvor, C.J., McLoone, M., McCanny, J.V.: Hardware elliptic curve cryptographic processor over GF(p). IEEE Trans. Circuits Syst. I, Reg. Papers 53(9), 1946–1957 (2006)
11. Kendall, A., Hamad, A., Daler, R.: Flexible Hardware Processor for Elliptic Curve Cryptography Over NIST Prime Fields. IEEE Transactions on Very Large Scale Integration Systems 17(8) (2009)
12. Haodong, W., Qun, L.: Efficient Implementation of Public Key Cryptosystems on Mote Sensors. In: Ning, P., Qing, S., Li, N. (eds.) ICICS 2006. LNCS, vol. 4307, pp. 519–528. Springer, Heidelberg (2006)