ISSN: 2278 – 909X
International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 7, Issue 4, April 2018

LOW POWER CRYPTOGRAPHIC PROCESSOR DESIGN FOR PORTABLE MOBILE DEVICES SECURITY

[1]Anitta George, Ravikumar M
[1]Pg scholar, Department of ECE, Mahendra Engineering College, Namakkal District, T.N, India
[2]AP, Department of ECE, Mahendra Engineering College, Namakkal District, T.N, India

***Abstract* --Until recently, the most widely supported public key encryption scheme was RSA, but now there's a new kid on the block -- Elliptic Curve Cryptography (ECC) -- which is set to become the leading next-generation public key cryptosystem. The reason for the excitement surrounding ECC it that it requires much smaller keys than RSA to provide the equivalent security; also, ECC is extremely computationally efficient providing savings in terms of time, memory, bandwidth, and energy consumption. In the case of the hand-held product arena, personal digital assistant (PDA) and communication devices simply don't have the processing capability to use RSA keys of 3,072 bits and higher. The solution is to use ECC, which requires much less processing while at the same time being much harder to crack. I introduce a new technique for point addition in affine coordinate which requires fewer registers. Based on this technique, I propose extremely small processor architecture for scalar multiplication. Through Field Programmable Gate Array (FPGA) implementations, I evaluate the area, performance, and energy consumption of the proposed crypto-processor. Utilizing two different working frequencies, it is shown that the proposed architecture reaches better results compared to the previous works, making it suitable for extremely-constrained, secure environments.**

I.INTRODUCTION

Elliptic curve cryptography is rapidly become the standard for public key ciphers because of the large amount of security provided per key bit. Several accreditation bodies have migrated to ECC for their public-key cryptographic requirements. To match the speed requirements for real time applications hardware acceleration of ECC is a necessity.For the inversion another important field primitive, the ItohTsujii algorithm (ITA), is designed to use optimal exponentiation circuits specific to the LUT size of the underlying FPGA platform. These field primitives are combined to realize the elliptic curve scalar multiplier. This project explores opportunities for pipelining the design for high-speed. This project extends the work proposed in to approximate the delay in the critical path by the number of k input LUTs in the path. This is used to determine analytically the location of the pipeline stages in the architecture. The number of pipeline stages in the architecture also critically affects the computation time. The optimal number of pipeline stages in the design.

The framework has been applied on two well-known scalar multiplication techniques, namely, left-to-right double and add algorithm with binary signed digit representation of the scalar, and the Montgomery ladder based scalar multiplication. The analysis shows that, while a three stage pipeline gives best performance for the former scalar multiplication method, a four stage pipeline is more suited for the latter.The three- stage pipelined architecture for double and add based scalar multiplication on Xilinx Spartan 3E platforms has a computation time of 10 μs with an area of 3789 slices. Further, the same technique when applied on the Montgomery ladder based scalar multiplier, the four-stage pipelined architecture requires a computation time of 9.5 μs, with an area of 3513 slices on the Spartan 3E platform.

II.EXISTING SYSTEM

RSA Algorithm

Over the years, in order to achieve high level of security, key size has grown up to more than 1000 bits. Long key size gives a high level of security, but it requires computing power, storage and bandwidth. So it cannot be used in devices that have limited processing power and battery life. In 1978, Ron Rivest, Adi Shamir and Len Adelman prepared RSA public key cryptography at MIT. It was the most popular public key algorithm over the past twenty years. RSA is an encryption algorithm that is used to encrypt messages, create digital signatures. RSA is based on integer factorization problem.

Disadvantages of Existing System

In cryptography, the RSA problem summarizes the task of performing an RSA private-key operation given only the public key. The RSA algorithm raises a message to an exponent, modulo a composite number N whose factors are not known. As such, the task can be neatly described as finding the eth roots of an arbitrary number, modulo N. For

large RSA key sizes (in excess of 1024 bits), no efficient method for solving this problem is known; if an efficient method is ever developed, it would threaten the current or eventual security of RSA-based cryptosystems—both for public-key encryption and digital signatures.

More specifically, the RSA problem is to efficiently compute P given an RSA public key (N, e) and a cipher text $C \equiv Pe \pmod N$. The structure of the RSA public key requires that N be a large semi prime (i.e., a product of two large prime numbers), that $2 < e < N$, that e be co prime to $\varphi(N)$, and that $0 \leq C < N$. C is chosen randomly within that range; to specify the problem with complete precision, one must also specify how N and e are generated, which will depend on the precise means of RSA random key pair generation in use.

As of 2010, the most efficient means known to solve the RSA problem is to first factor the modulus N, which is believed to be impractical if N is sufficiently large (see integer factorization). The RSA key setup routine already turns the public exponent e, with this prime factorization, into the private exponent d, and so exactly the same algorithm allows anyone who factors N to obtain the private key. Any C can then be decrypted with the private key.

By the above method, the RSA problem is at least as easy as factoring, but it might well be easier. Indeed, there is strong evidence pointing to this conclusion: that a method to break the RSA method cannot be converted necessarily into a method for factoring large semi primes.

In addition to the RSA problem, RSA also has a particular mathematical structure that can potentially be exploited without solving the RSA problem directly. To achieve the full strength of the RSA problem, an RSA-based cryptosystem must also use a padding scheme like OAEP, to protect against such structural problems in RSA.

III.PROPOSED SYSTEM

This Project presented the construction of an elliptic curve scalar multiplier for high-speed and area-constrained applications. A theoretical model was used to analyze delay of critical paths in the ECSMA and to obtain optimal pipelining. Efficient choice of scalar multiplication algorithm, optimized field primitives, balanced pipeline stages, and enhanced scheduling of point arithmetic resulted in a high-speed architecture with a significantly small area.The purpose of the hardware implementation is to give some common platform and fair comparison between our proposed architecture and similar previous designs. The focus in this study is not targeted toward the details of the architecture implementation; instead our aim is to extract the hardware time and area parameters of the main blocks to build a fair comparison study between the designs.

Therefore, our implementation exploration here is going to be limited to the level needed to serve this comparison goal.A group structure used to implement the cryptographic schemes is provided by using Elliptic curves and is determined over a finite field. The elements of the group are the points on the elliptic curve. They act as the identity element of the group. On the other hand the group operation can be executed by arithmetic operations based on finite field. It is discussed in detail in the next section.

Advantages of Proposed System

Now, tags become essential applications. They are used to categorize goods or give an access for an individual to categorize books in libraries and in other areas. An RFID system consists of a wireless reader device to communicate with an RFID tag. These tags are important in helping to identify counterfeit products out of the original. The issue raised is the possibility of copying these tags. By eavesdropping on the channel between the chip and the reader, the counterfeiter can create another chip with the same information of the original hold. It means that there is no protection on data.The systems that use symmetric protocol for authentication require the security of the secret key stored in the reader. The connections to the database also need to be secured in order, to prevent the system from begin compromised. This protection is very expensive to be provided.

IV.KARATSUBA ALGORITHM

The Karatsuba algorithm is a fast multiplication algorithm. It was discovered by Anatoly Karatsuba in 1960 and published in 1962. It reduces the multiplication of two n-digit numbers to at most single-digit multiplications in general (and exactly when n is a power of 2). It is therefore faster than the classical algorithm, which requires n2 single-digit products. For example, the Karatsuba algorithm requires 310 = 59,049 single-digit multiplications to multiply two 1024-digit numbers (n = 1024 = 210), whereas the classical algorithm requires (210)2 = 1,048,576.

The Karatsuba algorithm was the first multiplication algorithm asymptotically faster than the quadratic "grade school" algorithm. The Toom–Cook algorithm is a faster generalization of Karatsuba's method, and the Schönhage–Strassen algorithm is even faster, for sufficiently large n.FPGA-Based Elliptic Curve Cryptography Co-ProcessorElliptic curve cryptosystems are public key protocols whose security is based on the conjectured difficulty of solving the discrete logarithm problem on an elliptic curve.

Assuming Q to be a point of order n on an elliptic curve it is desirable to compute mQ, where m is an integer smaller than n. This will be done by using several additions, doublings, or possibly negations of points on the elliptic curve to achieve the result.These operations boil down to arithmetic operations in the finite field K = Fqn, over which the elliptic curve has been defined. In this work we concentrate on fields which have characteristic 2, i.e., q is a power of 2.The required computations to compute mQ can be categorized at three levels. Each requires thorough investigations to enable the design of a high performance elliptic curve co-processor.

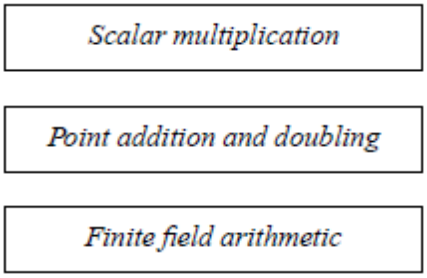


Figure 3.4 Three Stages of Performing Elliptic CuveHardware Implementation

The purpose of the hardware implementation is to give some common platform and fair comparison between our proposed architecture and similar previous designs. The focus in this study is not targeted toward the details of the architecture implementation; instead our aim is to extract the hardware time and area parameters of the main blocks to build a fair comparison study between the designs. Therefore, our implementation exploration here is going to be limited to the level needed.

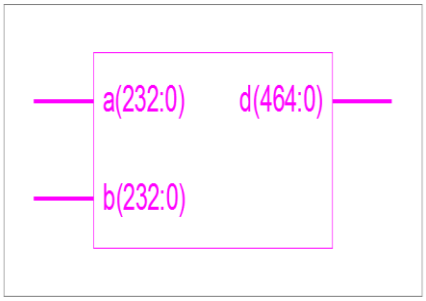
V.RESULTS AND DISCUSSIONS

The prototype of all the field arithmetic units and the scalar multiplier is realized in FPGA. For the hardware description Verilog HDL is used. Synthesis is done on the target device Xilinx XC2V2000. To show the effectiveness of the hardware acceleration the scalar multiplication is also implemented in software.MIRACL, an efficient cryptographic library is used to serve the purpose. All the codes were compiled using Visual Studio 6.0 and performance is measured on a Pentium IV 2.8 GHZ and 2GB RAM computer.Field multiplier is synthesized for different digit sizes. The resource utilization and the maximum frequency in which the multiplier runs are summarized Results for Scalar multiplication

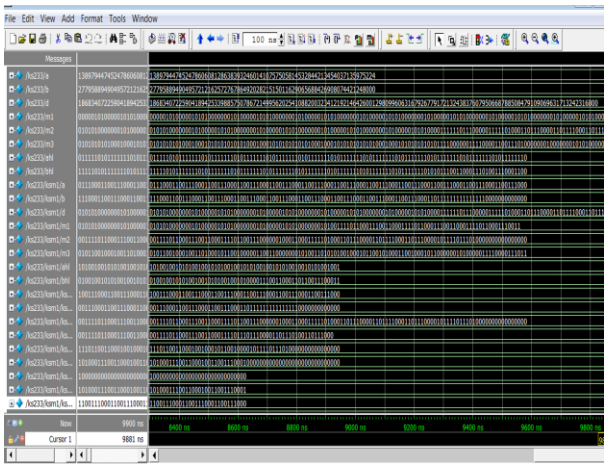
Table 5.3 performance and resource utilization for scalar multiplier

Multiplier digit size	Max freq (MHZ)	Latency for Kp (us)	#CLB slices	#Flip Flops	#LUT
1	203.421	298.1	1679(15%)	1393(6%)	3357(16%)
2	156.624	147.8	2178(21%)	1237(6%)	4117(19%)
4	99.745	123.3	4483 (42%)	1323(6%)	8609(40%)
6	173.317	89.5	4191(39%)	1643(8%)	8006(3%)
8	96.832	80.3	6660(62%)	1433(6%)	12588(59%)
12	166.325	46.7	7300(67%)	1918(9%)	14527(68%)

Here, 2.8GHz clock speed and 2GB RAM. This means the FPGA accelerated the point multiplication by 161 fold. Our result is also compared with other works and it is reported in Table 5-4. The slowest implementation among this works is Orlando & Parr's design . Among all the works J. Luaz' design is the most efficient in resource utilization and a single scalar multiplication takes 75µs. However, this work uses encoding for the scalar multiplier. However, it is the worst in resource utilization. This is due to allocation of separate multiplier for all the units in the design hierarchy.



Slice Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	512	19,200	2%
Number used as logic	512	19,200	2%
Number of occupied Slices	128	4,800	2%
Total equivalent gate count for design			3,968



Power summary:	I(mA)	P(mW)
Total estimated power consumption:		284
Vccint 1.00V:	174	174
Vccaux 2.50V:	43	108
Vcco25 2.50V:	1	3
Clocks:	0	0
Inputs:	1	1
Logic:	0	0
Outputs:		
Vcco25	0	0
Signals:	0	0
Quiescent Vccint 1.00V:	172	172
Quiescent Vccaux 2.50V:	43	108
Quiescent Vcco25 2.50V:	1	3

CONCLUSION

RSA is the most widely used public key cryptosystem. Recently, however, ECC is becoming the most prominent one. This is so because ECC is efficient for software as well as hardware realization. In addition, it gives a better security with shorter bit length than RSA. In this Project, hardware is designed and realization is done on FPGA for both curve and arithmetic operations. First we tried to implement efficient hardware for digit serial finite field multiplier based on the works. Then other field arithmetic operations are designed: squaring and Inversion. While realizing the inversion idle cycles are utilized to execute squaring resulting in a better implementation than the best result reported in the literature.After completing the finite field design, a scalar multiplier is designed.

We adopted Montgomery scalar multiplication algorithm . Modifications are made on this algorithm to efficiently realize it on hardware. Then the point doubling and point add operations are implemented using two multipliers, a squarer and XORs. These curve operations are implemented in parallel and the squaring and XOR is done in the idle cycle of the multiplication.

FUTURE SCOPE

The designed scalar multiplier is realized using Xilinx XCV2000 FPGA. A single point multiplication takes 47 micro seconds with resource utilization of 14527 LUTs and 1918 Flip Flops. This result is 161 fold faster than the software implementation and better in latency than the best result reported in the literature utilizing half of the resources used in the latter. There are other tasks for further work. The first one is loading the binary of our program on a real FPGA. After this work is properly completed, estimate the actual performance of the Hardware Acceleration of ECC based algorithms. Finally, support for the rest of NIST curves could be tried on a single FPGA.

REFERENCES

[1] Steffen Peter and Peter Langendorfer, ―An Efficient Polynomial Multiplier in GF(2m) and its Application to ECC Designs‖, Design, Automation & Test in Europe Conference & Exhibition, pp.1-6,2007.

[2] M.M.Sandoval, C.F.Uribe, R. Cumplido and I.A.Badillo, ―An Area/Performance Trade-Off Analysis of a GF(2m) Multiplier Architecture for Elliptic Curve Cryptography‖,Elsevier- Computers & Electrical Engineering , vol 35, issue 1, pp. 54–58,Jan. 2007.

[3] Bijan Ansari and M. Anwar Hasan, ―High-Performance Architecture of Elliptic Curve Scalar Multiplication‖, IEEE Trans. on Computers, vol. 57, no. 11, pp. 1241 - 1245 2008.

[4] Chester Rebeiro and Debdeep Mukhopadhyay, ―High Performance Elliptic Curve Crypto-Processor For Fpga Platforms‖ , proceedings on 9th International Conference on Cryptology in India, pp 376-388,dec 2008.

[5] Rahila Bilal and M.Rajaram, ―High Speed and Low Space Complexity FPGA Based ECC Processor‖, International Journal of Computer Applications,vol. 8,no.3,pp.5-10,2008.