Journal of Theoretical and Applied Information Technology
15th August 2019. Vol.97. No 15

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

COST-EFFECTIVE FPGA IMPLEMENTATION OF PARALLEL SCHMIDT-SAMOA CRYPTOSYSTEM (SSC)

QASEM ABU AL-HAIJA[1], IBRAHIM MAROUF[2], MOHAMMAD M. ASAD[3], KAMAL AL NASR[4]

[1] Tennessee State University, Computer Information and Systems Engineering, Nashville, TN, USA

[2] King Faisal University, Department of Electrical Engineering, Hufof 31982, Ahsa, Saudi Arabia

[3] University of Bristol, Department of Electrical and Electronic Engineering, Bristol, BS8 1QU, UK

[4] University of Texas at San Antonio, Department of Computer Science, San Antonio 78249, TX, USA

[1]qabualha@my.tnstate.edu, [2]imarouf@outlook.com, [3]asammosab@gmail.com, [4]kamal.alnasr@utsa.edu

ABSTRACT

Schmidt-Samoa Cryptosystem (SSC) is a public key Cryptosystem. SSG is heavily based on modular arithmetic involving large prime number. In this paper, we consider a SSC that is used to secure the data over a communication system against vulnerabilities and attacks. We propose an efficient FPGA implementation of SSC cryptosystem that employs scalable arithmetic modules and effective number theory schemes in maximum parallelism exploitation. The provided simulation results show the advantage of using parallelization of system modules in optimizing the system performance of data security against attacks. On average, the encryption/decryption process of $128-$bit SSC registered a total delay of 125 ms approximately with maximum operational frequency of 40 MHz utilizing of 50% of the logic elements of Altera Cyclone IV FPGA EP4CGX22CF19C7 device and consuming 250 mW of thermal power. Consequently, the obtained results are attractive for FPGA designer of cryptographic algorithms with scalable design area, fast encryption/decryption processes and low power consumptions.

Keywords: *Cryptography, Computer Arithmetic, FPGA Design, Hardware Synthesis, Schmidt-Samoa Cryptosystem, Integer Factorization.*

1. INTRODUCTION

In recent years, advanced hardware design and synthesis have been achieved in the development of high density, low cost, high frequency, and enhanced power consumption of flexible and reconfigurable logic blocks with specific IP cores. In addition, they are used in the applications that have the ability of building different high precision embedded and digital processors. Such devices are considered as Field Programmable Gate Arrays (FPGAs) [1]. Due to this re-configurability and flexibility, FPGAs have attracted the hardware designers to implement a wide range of contemporary applications that are urgently on-demand such as the design and synthesize of public key cryptographic systems. Public key schemes [2] are preferred to use due to many reasons such as the non-existence of the secure communication channels. Schmidt- Samoa Cryptosystem (SSC) [3] is an example of a public key cryptosystem that can be used to secure data transmission over non-secure communication networks.

SSC is an asymmetric cryptographic technique that is significantly based on modular arithmetic involving large prime number used for data encryption and decryption. The usage of the large prime number and modular arithmetic is to provide different security services such as confidentiality, integrity, authentication, and non-repudiation. The security of SSC algorithm is considerably based on the difficulty of its integer factorization-problem in which an integer is decomposed to its product of smaller numbers (usually prime numbers). The

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

complexity in this method arises when factoring a very large number because there is no such known efficient algorithm. Although SSC is proved to be very secure [4], there is no such a perfect system. SSC is vulnerable to some known attacks such as Brute-force attack, Man-in-the-Middle attack, and Side Channel attack. Generally, all public key cryptography algorithms suffer from these attacks [3].

Since the efficient implementation of public key cryptosystems is on demand, especially if it implemented with appropriate hardware technology with high precision design, in this paper, we are reporting on the efficient FPGA design and implementation of SSC Cryptosystem using Altera Cyclone IV FPGA chip family with target device EP4CGX-22CF19C7. The completed design of SSC is comprised of seven components: random number generation, multiplication of the prime numbers, encryption key based on least common multiple, decryption key based on modular inverse, encryption and decryption processes using modular exponentiation. The synthesized experimental results show that SSC can be used as efficient and comparable alternative to RSA. RSA is a well-known secure and practicable public key scheme that can be used to protect information during the transmission over the insecure channels. To the best of our knowledge, the hardware design of SSC Cryptosystem, that maintains a maximum parallelism between its underlying computation modules to optimize the system performance factors, has not been investigated previously. Finally, this system can be utilized for several security claims such as network routers and security gateways using certain protocols. Specifically, the main contributions of this paper can be summarized as follows:

- We develop a detailed model and architecture for SSC that can maintain parallelism of its underlying modules to optimize the system performance in terms of several cost factors (such as, design delay, area and power consumption).
- We provide details on the hardware implementation of a 128-bit SSC coprocessor using efficient modules, including a schematic diagram, a finite state machine, timing waveform diagrams, and a sample simulation runs.
- We provide extensive simulation results to gain insight into the proposed model and the solution approach. Discussion of the synthesize results related to the area of the design, the total delay of the design, minimum delay, maximum frequency and total FPGA thermal power dissipation complexities.

The rest of this paper is organized as follows. Section II describes the proposed system model and design architecture for the parallel implementation of SSC. Section III provides the proposed implementation approach and environment as well as cost factors explanations. Section IV presents and discusses the simulation results by considering several design modules. Finally, Section V concludes the paper.

2. SYSTEM MODELING ARCHITECTURE

To start a secure communication session, the receiver, who is Alice in this case, starts by choosing two large prime numbers (p, q) and then compute her public key $N = p^2 q$. Alice then share the public key (N) with Bob (and even other senders) who will use it to encrypt the plaintext messages communicated with Alice. Again, Alice computes her private key (d) to be used for decryption processes $d = N^{-1}$. Next, using the private key, Alice decrypts the cipher-text. The complete SSC algorithm diagram is illustrated in Figure 1, where the SSC process is divided into three stages: key generation stage, Encryption stage, and Decryption stage. The challenge in SSC is the ability to factor out the public key, which is the product of two very large primes. As the size of the key increases, the factorization problem becomes even more complicated [2]. Factoring a number means defining that number as a product of prime numbers. In SSC, factoring the public key (N) means as breaking the cryptosystem. If an attacker can factor out the public key, he can easily calculate the private key (d) and decrypt any data. As public key $N = p^2 q$, is known to everyone, therefore factoring (N) leads to compute p and q. Then the private key can be computed using the congruent equation where LCM is the least common multiple of two numbers:

$$d \equiv N^{-1} \ mod \ \ LCM(p-1, q-1)$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

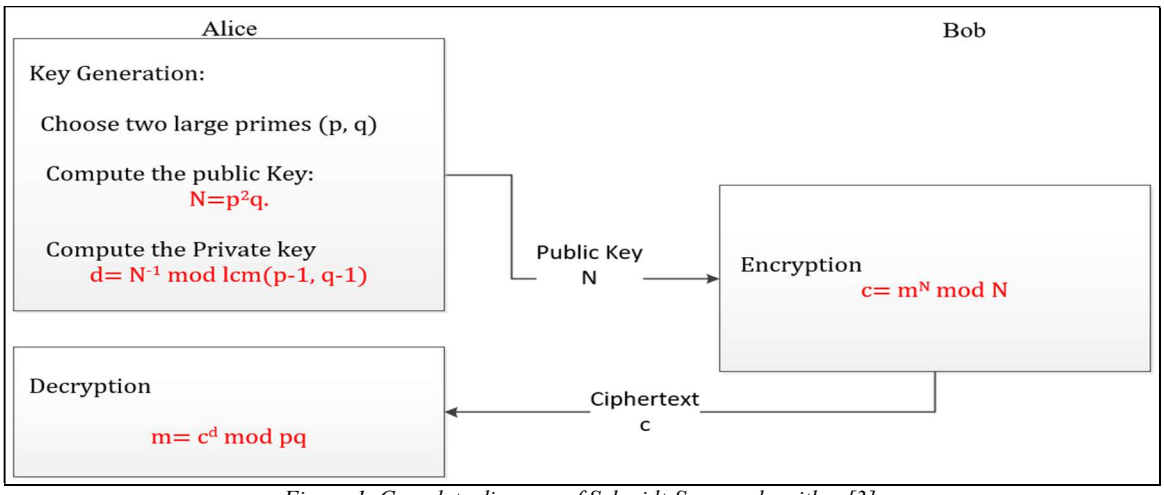


Figure 1. Complete diagram of Schmidt-Samoa algorithm [3]

The top view design of SSC Coprocessor is shown in Figure 2 (a) which clarifies the two modes of operations of SSC*: encryption and decryption*. In both modes, the operation starts by a clock trigger (falling edge clock in this case) and controlled by two more control signals at the input (i.e. enable and reset) and two control signals at the output (i.e., ack. and ready). The proposed design aims to achieve the maximum computing speed of SSC Crypto-processor by exploiting the maximum level of parallelism at both the programming level and hardware utilization and combination level. To accomplish the proposed design efficiently, we firstly divide the proposed SSC process into two main stages, namely: $SSC\_Initializing$ *stage* (since it will be performed once) and *encryption/decryption stage* (this is re-used with each secure communication session). These stages are illustrated in Figure 2(b), which clarify the simplified view of the proposed SSC Crypto-processor. After $SSC\_Initializing$ stage is performed, the public key 'N', the private key 'd', and the decryption modulus 'n' are generated, and, therefore, the encryption/decryption sessions are started. From the figure, SSC Initialization stage was designed with different alternatives whether to find the public key 'N' or the modulus value used for decryption 'n' first, keeping in mind the need for generating two large primes.

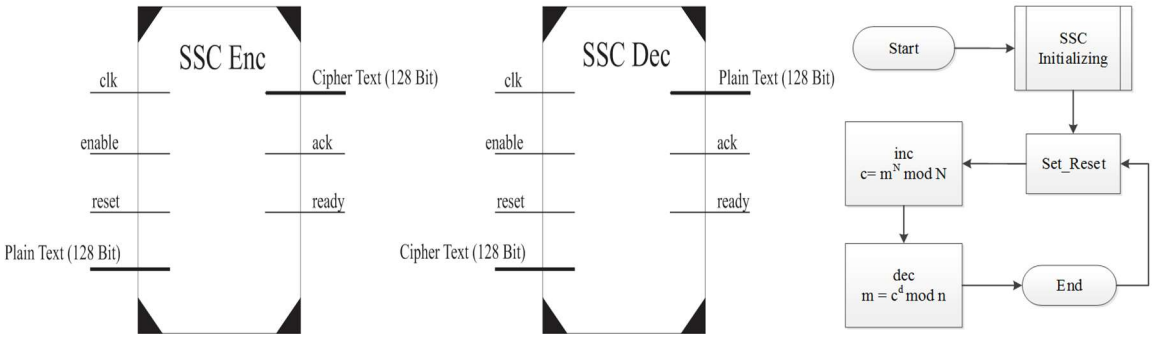
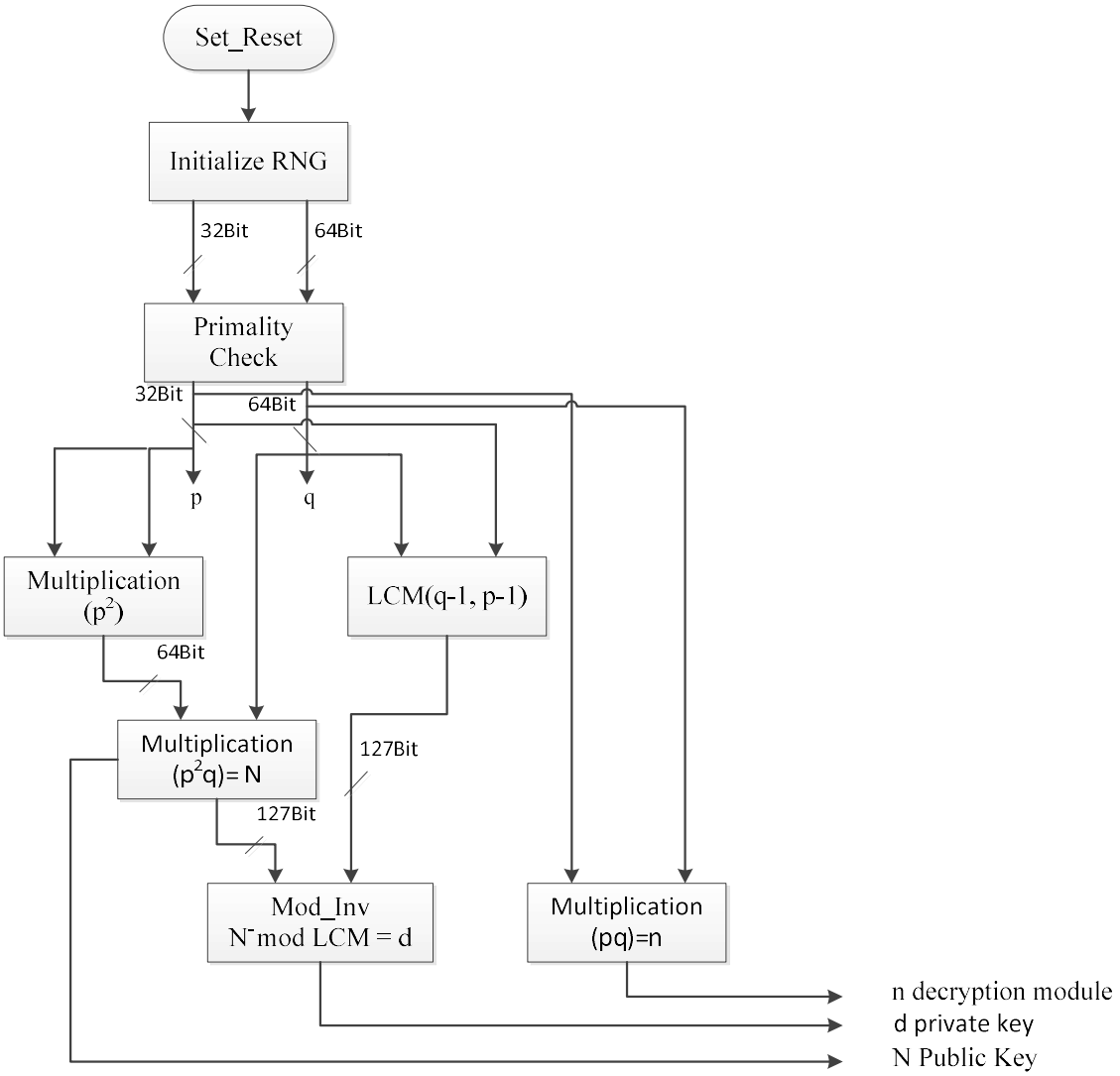


Figure 2. (a) The top view diagram of SSC (b) Simplified view of SSC processor

Figure 3 illustrates the detailed design along with finite state machine (FSM) diagram for the $SSC\_Initializing$ stage exploiting the maximum possible parallelism and pipelining of the SSC design modules. From the figure, the stage starts by initializing the RNG unit (random number generator), then the primality tester unit is initialized to generate the prime number p first. After 'p' is generated, the primality tester is started and generate the second prime 'q'. At the same time p^2 is computed through the multiplication unit (i.e., parallelism is achieved). Next, the public key $N = p^2q$ is computed by the same multiplication unit (i.e., area optimization) after the prime number 'q' is generated. Since generating primes takes longer time comparing to the multiplication operation (i.e., pipelining is achieved),

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

at the time the two prime numbers are decremented and since they are primes, they must be odd, thus decrementing the odd numbers in binary representation can be done in the fly by just wiring manipulations and setting LSB to zero (i.e., Speed optimization) and then sent to LCM unit to compute their least common factor parallelism is achieved). Finally, the private key $d = N^{-1} \, mod \, N$ and the decryption module $n = pq$ are computed in parallel.



ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

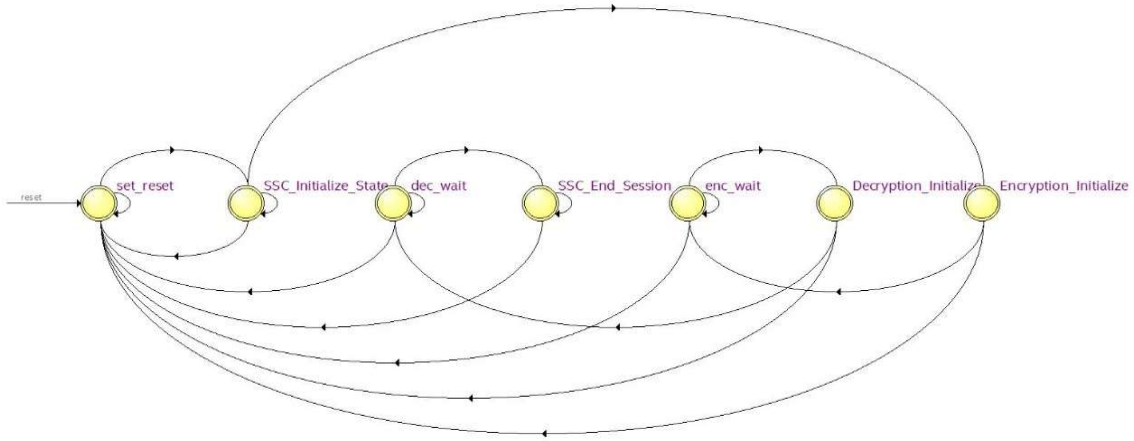


Figure 3. (a) Internal architecture of SSC_Initializing Stage (b) Finite State Machine of initialize SSC

Once the SSC initialization phase finished, the system is now ready to encrypt and/or decrypt using the generated credentials of SSC initialization as illustrated in Figure 4. Note that, both processes can be executed independently using parallel modular exponentiation units.

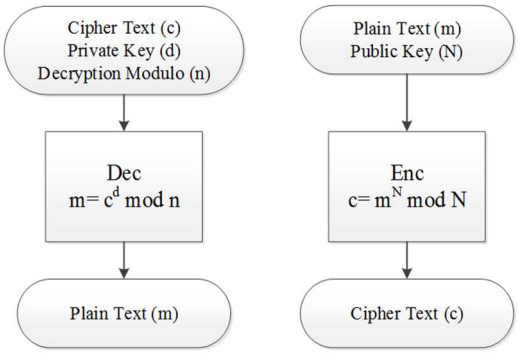


Figure 4. Post Initialization Phase: Encryption / Decryption Phases

The proposed design of SSC Crypto-processor is promising and expected to be a good alternative for many robust cryptosystems such as RSA which known to be impartially powerful Crypto-processor especially when it is implemented in hardware platform with large key sizes ($\geq$ 1 $Kbit$).

3. IMPLEMENTATION ENVIRONMENT AND COST FACTORS

To implement the proposed $FPGA$ design of SSC cryptosystem, we have used the $Altera\ Cyclone\ IV\ FPGA$ family [5] with target device of $EP4CGX22CF19C7$ using structural $VHDL$ [6] coding as a hardware description language along with $Quartus\ II$ [7] and $Modelism\ 10.1d$ [8] packages as platforms for simulation and synthesizing purposes. In addition, a high-performance multiprocessor computing platform has been used in the phases of coding, simulation, and verification as well as in synthesizing and testing. Furthermore, the critical path delay results were generated using $TimeQuest$ timing analyzer tool of $Quartus\ II$ software package with fast $1200mV\ 0C$ model and area estimation results were generated using *analysis and synthesize tool* after post-fitting mapping and port assignments.

3.1 Target Cost Factors

In this work, we considered three cost factors to specify the efficiency of the proposed design, namely: the design delay time, the design area and the thermal power dissipation of the $FPGA$ design. We have utilized the estimation tools of Quartus II software to generate the corresponding reports of such factors.

Estimation of Design Delay Time: Timing analysis measures the delay of a signal reaching a destination in a synchronous system. This is accomplished using ***TimeQuest*** Analyzer of Quartus II simulation package after the Quartus II Fitter optimizes the placement of logic in the device in order to meet timing constraints. ***TimeQuest*** tool analyzes the timing paths in the design, calculates the propagation delay along each path, checks for timing constraint violations, and reports timing results. To illustrate the delay time calculation, we provide an example of time delay waveform generated by ***TimeQuest*** Analyzer in Figure 5. From the figure, we can extract the following timing values:

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$The\ clock\ arrival\ time\ (clock\ delay)$$

$$= 0.663\ ns$$

$$Data\ Delay\ (combinational\ logic\ delay)y)$$

$$= 2.866\ ns$$

$$The\ total\ data\ arrival\ delay$$

$$= 0.663\ +\ 2.866\ =\ 3.529\ ns$$

$$Slack\ time = +5.471\ ns$$

Note that the *slack time* [7] is the margin by which a timing requirement was whether met or not met. If it is a positive value that means the timing requirement is met. If its value is negative, then the next clock comes before the data has arrived and it indicates error. In this wave, we see that there is a plenty of free time.

Estimation of Design Area: The design area of Altera FPGA devices can be defined as the total number of logic elements (LEs) that are utilized to accomplish the target design. For Cyclone IV E family device that is used in this paper, every logic element consists of a four-input LUT, a programmable register, and a carry chain. The number of logic elements of the proposed FPGA design was measured by ***Quartus II Synthesizer tool*** and was reported as the total number of LEs. However, the maximum number of LUTs (i.e., Area of FPGA) can be varied according to the density of the chip technology as illustrated in Figure 6 (a) [9]. It can be clearly seen that increasing the FPGA density (i.e. decreasing the Nano-meter size) allows more LUTs to be included in the chip technology.

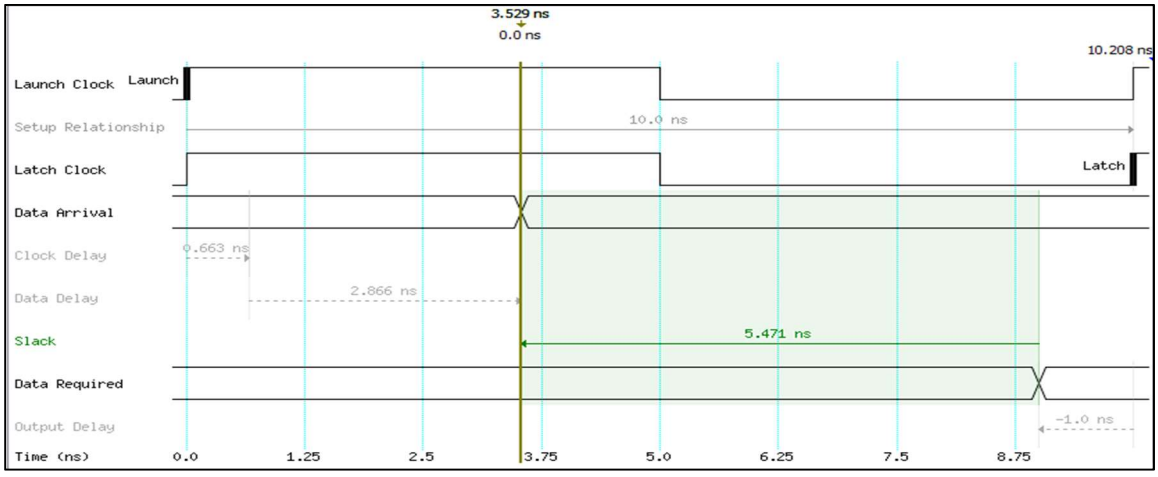
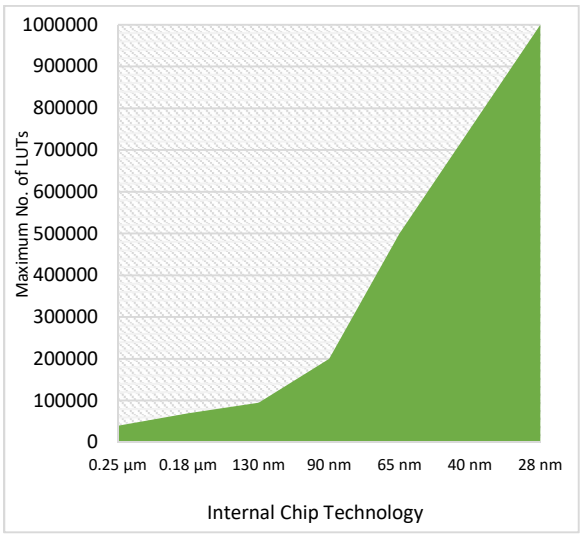


Figure 5. A waveform sample for time delay calculation

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



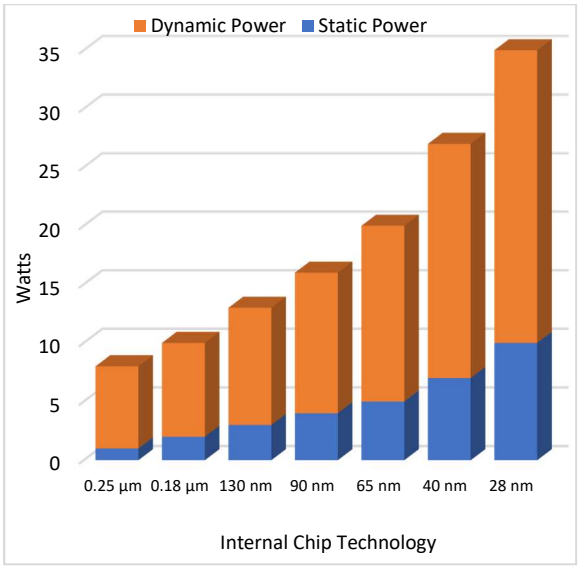


Figure 6. (a) Maximum No. LUTs vs FPGA density in nm b) Power consumption in Watts vs FPGA density in nm

Estimation of Design Power Consumption: The power dissipation of the FPGA design using Altera devices [9] can be defined as the total calculated power of three sources: (1) Core dynamic thermal power dissipation (charging and discharging capacitance on internal node). (2) Core static thermal power dissipation (standby power). (3) I/O thermal power dissipation (charging and discharging external load capacitance connected to device pins). The total thermal power dissipation of the proposed FPGA design was estimated by *PowerPlayAnalyzer* [10]. Power Analyzer is used to produce a power consumption profile representative of the design implementation after fitting. The power consumption can be varied according to the Nano-meter of the chip technology. Figure 6 (b) shows the percentages of FPGA power was consumed in static and dynamic modes [9]. It can be clearly seen that increasing the FPGA density (i.e. decreasing nm size) will require more thermal power.

3.2 Design Constraints

Altera FPGA Cyclone IV E supports up to 512 I/O ports. 14 ports are used for control signals, the remaining 498 ports can be initialized as either input/output or buffer as desired. Thus, 1K-bit SSC cannot be implemented using this type of FPGA devices. Further, the 256-bit SSC cannot be implemented since 512 ports are required in this case for I/O ports. Thus, the appropriate choice is a 128-bit SSC. Moreover, the high precision design-based Microcontroller was not implemented due to many reasons. The main reason is the program execution time. It takes very long time since high precision data size is proposed. By simple calculations, generating the private key in SSC might take days. Another reason is that SSC must be implemented in such a way that involves defining look up table and memory.

4. EXPERIMENTAL RESULTS

SSC is a public key cryptosystem in which its computations are significantly based on the use of several digital arithmetic and modular arithmetic algorithms as well as different number theory schemes. It employs the properties of prime numbers alongside the congruent to produce a very secure hard to break cryptosystem. Arithmetic operation like multiplication and squaring, and modular exponentiation and modular inverse are involved in the algorithm to add complexity to the cipher. Consequently, to accomplish the proposed robust

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SSC Crypto-processor, we have selected the underlying algorithms and theories to be scalable and efficient to build. The aim of our SSC Crypto-processor is to optimize the performance of the system. All experimental results are generated using Quartus II tools along with Altera ModelSim. In this work, we have performed extensive experimental traces for several cost factors and for almost all design components. In this section, we summarize the results of this substantial number of experiments in Table I for each of the implemented algorithms.

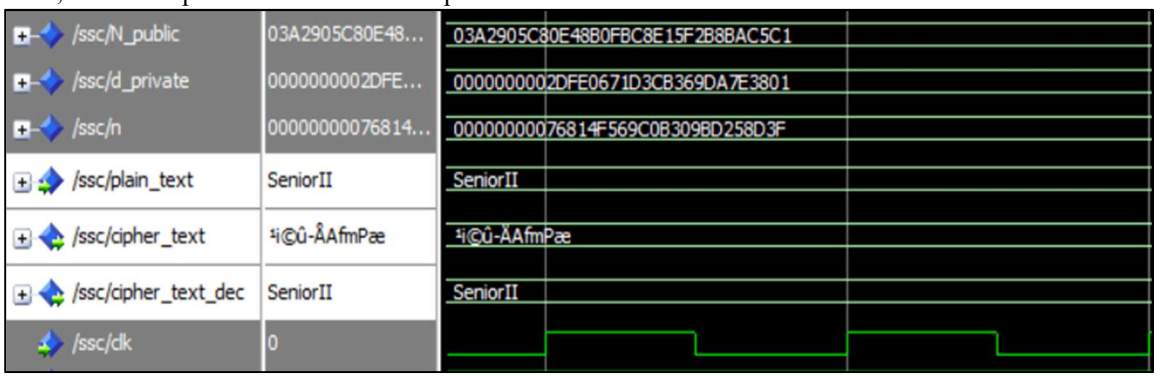


Figure 9. Run simulation of SSC Encryption process using ModelSim

Eventually, we have implemented the parallel scheme of 128-bit SSC as discussed earlier in section II using the modules shown in Table I. The proposed design has been synthesized using Quartus II tool targeting Altera Cyclone IV EP4CGX22CF19C7 FPGA device and simulated using $ModelSim$ 10.1 simulator tool to verify the functionality of the SSC Crypto processor.

Figure 9 provides a sample numerical simulation run of SSC encryption process including SSC parameters required to encrypt the plain message *"SeniorII"*. The resultant cipher text is shown in the figure where p, q are generated randomly and tested to ensure they are primes. Then the generated p, q are used to compute N_public, $d\_private$, and encryption modular N.

Table 1: Summary of cost factors for the SSC underlying units

Module	Algorithm (s)	Clock Cycle (ns)	Maximum Frequency (MHz)	Design Area ($\#LEs$)	Power Dissipation (mW)
Random Number Generation (RNG)	Hybrid Two Stage RNG (Trivium + LFSR) [11] [12]	7.659	217.39	656	164.65
Primality Testing (PT)	Millar-Rabin [13] [14]	22.646	51.11	6184	151.29
Conventional Addition	Kogge-Stone Adder (KSA) [15]	4.504	284	569	143.11
Redundant Addition	Carry Save Adder (CSA) [16]	3.529	349	129	32.44
Arithmetic Multiplication	Wallace Tree CSA based Radix-8 Booth multiplier [17] [18]	14.103	90.83	14249	217.56
Least Common Multiple (LCM)	GCD reduction method [19]	12.131	110.67	5122	179.80
Greatest Common Devisor (GCD)	Pulse-Minus GCD [20] [21]	10.17	140.88	1,810	205.53
Modular Exponentiation	Right to Left Modular Exponentiation [22] [23]	25.521	44.59	2556	221.82
Modular Inverse	Extended Euclidian Modular Inverse [24] [25]	16.845	72.7	7157	231.41

Moreover, we have analyzed the performance of the proposed SSC on three cost factors $(Area, Time, Power)$ by using the hardware synthesis tools of $Quartus\ II$ simulation and verification package. The hardware synthesis results show that the total number of employed logic elements (LEs) is 58719 LEs along with 29883 registers is recording a 50% of hardware

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

utilization of the FPGA kit resources. In addition, the total number of pins is 389 pins out of 512 $pins$ utilizes about 76% of $total\ I/O$ pins available in the target FPGA kit. On other hand, regarding the timing analysis, the total path delay for the critical clock cycle as recorded at 25.02 ns and a maximum frequency 40 MHz. The timing waveform analysis is shown in Figure 11. Moreover, to calculate the average total delay of SSC Cryptosystem, we have run the simulation several times to count the number of clock cycles spent on encryption/decryption process. On average we have got almost 5 million clock cycles to perform SSC process which in turn registered a total processing delay of 125 ms approximately.

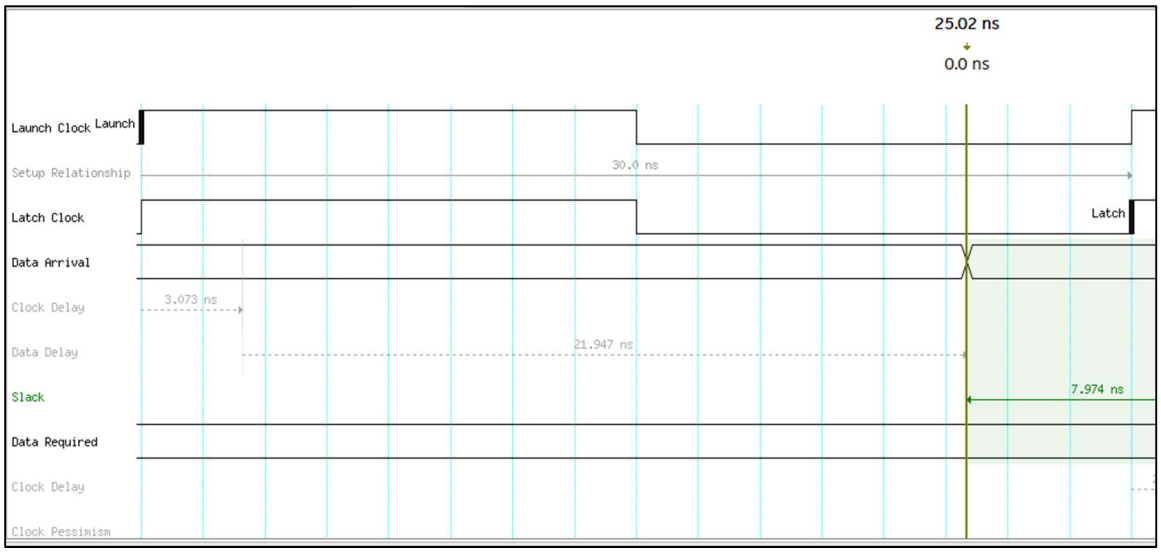


Figure 11. A time waveform for SSC delay for critical Clock Cycle

Finally, the proposed 128-bit SSC was also synthesized to target the thermal power dissipation. It was found that the proposed design consumes almost 251.3 mW of total FPGA thermal power in which are distributed as 102.6 mW of static power and 148.7 $mW\ of\ I/O$ assignments and operations (dynamic power).

5. CONCLUSIONS AND REMARKS

SSC is a powerful PKC algorithm based on the hardness of factoring large numbers problem. In this paper, we have thoroughly discussed SSC underlying designs from many aspects: maximum frequency and critical path delay, design area, power consumption, maximum parallelism achieved and design correctness. The proposed hardware cryptosystem design is conducted using Altera Cyclone FPGA design technology along with the help of CAD package of Altera such as Quartus II and Modelsim 10.1. The proposed parallel mplementation showed an attractive results in terms of cost factors recorded as: total delay of 125 ms, maximum frequency of 40 MHz, 8719 logic elements-LEs (50% of the logic elements LEs of the used FPGA kit) and 251.3 mW of total FPGA thermal power dissipation. To sum up, a synthesis of the efficient SSC cryptographic security-based system was successfully implemented via FPGA technology. The design and implementation were executed for 128-bit encryption and decryption keys. We have observed that the message that will be encrypted should be limited to the range of the modulus. This implies that the range should begin from zero to the modulus minus one.

REFERENCES

[1]. C. Maxfield, “The Design Warrior’s Guide to FPGAs: devices, tools and flows”, Mentor Graphics Corporation and Xilinx., Elsevier, 2004

[2]. C. Paar, J. Pelzl, (2010) ‘Understanding Cryptography’. Springer-Verlag Berlin Heidelberg Publisher. https://doi.org/10.1007/978-3-642-04101-3.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[3]. Q. Abu Al-Haija, M. M. Asad, I Marouf, "A Systematic Expository Review of Schmidt-Samoa Cryptosystem", International Journal of Mathematical Sciences and Computing (IJMSC), Vol.4, No.2, pp.12-21, 2018.DOI: 10.5815/ijmsc.2018.02.02

[4]. K. S. Samoa, (2006) 'A New Rabin-type Trapdoor Permutation Equivalent to Factoring', Electronic Notes in Theoretical Computer Science, Elsevier, vol.157, issue 3, p.p.79-94. https://eprint.iacr.org/2005/278.pdf

[5]. Altera Corporation, "Cyclone IV Device Handbook", Vol. 1, CYIV-5V1-2.2, https://www.altera.com/, 2012

[6]. B. J. LaMeres, "Introduction to Logic Circuits & Logic Design with VHDL", Electronics & Electrical Engineering, Springer 2017.

[7]. Altera Corporation, "Introduction to Quartus II Software: Ver 10.0", Intel Quartus II MNL-01055-1.0, 2012.

[8]. Altera Corporation, "Simulation Quick-Start for ModelSim - Intel FPGA Edition", Intel Quartus Prime Standard Edition UG-01102, 2017.

[9]. Achronix Semiconductor Corporation: https://www.achronix.com/

[10]. Intel Corporation, "PowerPlay Early Power Estimator User Guide", UG-01070, 2017.02.21

[11]. Q. Abu Al-Haija, I. Marouf, M. M. Asad, "A Double Stage Implementation for 1-K Pseudo RNG using LFSR and TRIVIUM", Journal of Computer Science and Control Systems (JCSCS), University of Oradea Publisher, Vol. 11, No. 1, May 2018

[12]. Y. Tian, Chen, G. and Li, J, "On the design of Trivium", Beijing Daxue Xuebao Ziran Kexue Ban/Acta Scientiarum Naturalium Universitatis Pekinensis, Vol. 5: 431, 2009. Available at http://eprint.iacr.org/2009/43.

[13]. M. M. Asad, I. Marouf, Q. Abu Al-Haija, "Investigation study of Feasible Prime Number Testing Algorithms", ACTA TECHNICA NAPOCENSIS - Electronics and Telecommunications, Technical University of Cluj-Napoca, Vol. 58, No. 3, 2017

[14]. S. Ishmukhametov, B. Mubarakov, "On practical aspects of the Miller-Rabin Primality Test", Lobachevskii Journal of Mathematics, Vol. 34, No. 4, Pp. 304–312, 2013.

[15]. I. Marouf, M. M. Asad, A. Bakhuraibah, Q. Abu Al-Haija, "Cost Analysis Study of Variable Parallel Prefix Adders Using Altera Cyclone IV FPGA Kit", IEEE International Conference on Electrical & Computing Technologies & Applications, (ICECTA), 2017.

[16]. M.D. Ercegovac, and T. Lang, Digital Arithmetic., Morgan Kaufmann Publishers, Elsevier: San Antonio, USA, vol. 1. 2004.

[17]. M. M. Asad, I. Marouf, Q. Abu Al-Haija, "Review of Fast Multiplication Algorithms for Embedded Systems Design", International Journal of Scientific & Technology Research (IJSTR), IJSTR Publishing, Vol.6, No.8, 2017.

[18]. M. M. Asad, I. Marouf, Q. Abu Al-Haija, "Radix-8 Design Alternatives of Fast Two Operands Interleaved Multiplication: with Enhanced Architecture With FPGA implementation & synthesize of 64-bit Wallace Tree CSA based Radix-8 Booth Multiplier", International Journal of Advanced Network, Monitoring and Controls, Exeley Inc. Publishing, Volume 04, No.02, pages 15-27, 2019

[19]. W. Stein, "Elementary Number Theory: Primes, Congruence, and Secrets: A Computational Approach", Number Theory and Discrete Mathematics, Springer- Verlag New York, Vol. 1, 2009.

[20]. I. Marouf, M. M. Asad, Q. Abu Al-Haija, "Reviewing and Analyzing efficient GCD/LCM Algorithms for Cryptographic Design", International Journal of New Computer Architectures and their Applications (IJNCAA), Society of Digital Information and Wireless communication (SDWIC), Vol.7, No.1, 2017

[21]. R. P. Brent and H. T. Kung. Systolic VLSI arrays for polynomial GCD computation. IEEE Transactions on Computers, C-33(8):731–736, August 1984.

[22]. I. Marouf, M. M. Asad, Q. Abu Al-Haija, "Comparative Study of Efficient Modular Exponentiation Algorithms", An International Journal of Advanced Computer Technology (IJACT), COMPUSOFT, Vol. 6, No. 8, 2017

[23]. C. D. Walter, "Right-to-Left or Left-to-Right Exponentiation?", 1st International Workshop on Constructive Side-Channel Analysis and Secure Design, Darmstadt, Germany, February 4-5, 2010.

[24]. Q. Al-Haija, A. AlShuaibi, A. Al Badawi, "Frequency Analysis of 32-bit Modular Divider Based on Extended GCD Algorithm for Different FPGA chips", International Journal of Computers & Technology (IJCT), CIRWORLD Publishing, Vol. 17, No.1, 7133-7139. https://doi.org/10.24297/ijct.v17i1.6992

[25]. J. Hlaváč and R. Lórencz. Arithmetic Unit for Computations in GF(p) with Left-Shifting Multiplicative Inverse Algorithm, Architecture of Computing Systems, ARCS 2013. Lecture Notes in Computer Science, vol 7767. Springer, 2013.