Analysis and Inner-Round Pipelined Implementation of Selected Parallelizable CAESAR Competition Candidates

Sanjay Deshpande and Kris Gaj
Electrical and Computer Engineering Department
George Mason University
4400 University Drive, Fairfax, VA 22030, U.S.A.
{sdeshpan, kgaj}@gmu.edu

Abstract— **In this paper, we have first characterized candidates of the Competition for Authenticated Encryption, Security, Applicability, and Robustness (CAESAR) from the point of view of their suitability for parallel processing of multiple blocks of associated data, message, and ciphertext. Then, we have chosen seven candidates from the Round 2 and Round 3 submissions, namely SCREAM, AES-COPA, Minalpher, OCB, AES-OTR, COLM, and Deoxys. We first obtained the initial estimates of the maximum clock frequency, throughput, area, and critical path for the high-speed Basic Iterative Architecture of each of the above candidates. Then, we implemented a two-stage inner-round pipelining for all the aforementioned algorithms in order to improve the frequency and throughput by reducing the critical path and processing multiple blocks of data simultaneously. We targeted the largest available FPGA in the student version of Xilinx ISE, i.e., Xilinx Virtex 6 XC6VLX75T-3FF784. Our results have demonstrated the improvement in the clock frequency and throughput by a factor varying from x1.28 for OCB to x1.84 for SCREAM, and the change in the throughput to area ratio (with area expressed using LUTs) by a factor varying from x0.93 for Minalpher to x1.72 for SCREAM.**

Keywords: cryptography, encryption, pipelining, FPGA.

I. Introduction

Authenticated Encryption (AE) or Authenticated Encryption with Associated Data (AEAD) is a cryptographic algorithm that simultaneously provides confidentiality, integrity, and authentication of message. The authenticated ciphers take message, associated data AD, a public message number Npub, and an optional secret message number Nsec as an input and generate resulting ciphertext C, tag T, and optional encrypted Nsec. The tag is appended to the end of the ciphertext to assure the integrity and authenticity of the transaction, as shown in Fig. 1. Decryption and tag verification are conducted in a similar fashion. Tag' is computed as above, and compared against the concatenated Tag. If Tag = Tag', then authentication and integrity of the transaction are assured; otherwise the decrypted ciphertext is not released. If authenticity and integrity are verified, the outputs of the transaction are the AD, message, and optional decrypted Nsec.

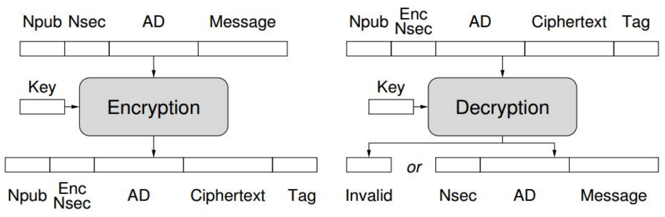


Figure 1. Input and Output of an Authenticated Cipher. Notation: Npub - Public Message Number, Nsec - Secret Message Number, Enc Nsec - Encrypted Secret Message Number, AD - Associated Data [16].

Cryptographic competitions have become a common way of developing cryptographic standards. This process has worked really well in case of Advanced Encryption Standard (AES), developed in the period 1997-2001, and then SHA-3 competition (Secure Hash Algorithm 3), conducted in the period 2007-2012. In 2013, a new contest, called CAESAR - Competition for Authenticated Encryption: Security, Applicability, and Robustness - has been announced. Each algorithm in the contest has been evaluated based on multiple criteria, including security, software and hardware efficiency, flexibility, simplicity, and any licensing encumbrances. The contest started off with 57 candidates in Round 1, and then reached Round 2 with 29 candidates, and Round 3 with 15 candidates remaining [7].

In this paper, we have analyzed all CAESAR Round 2 and Round 3 candidates from the point of view of their capability for parallel processing of blocks belonging to the same associated data, message, and ciphertext. Eleven Round 2 and five Round 3 candidates have been shown to have such capabilities, as summarized in Tables A1 and A2 in Appendix A. The further downselection was based on the maximum clock frequency of their high-speed Basic Iterative Architecture, reported in [5]. The ciphers with the lowest maximum clock frequency were selected for pipelining.

II. General Methodology

Pipelining is one of the well-known techniques used to increase the speed of any digital design. In this project, we have implemented inner-round pipelining of seven different authenticated ciphers. The inner-round pipelining provides substantial increase in the speed of the cipher, with the small increase in the circuit area [12, 13]. In this method, pipeline registers are inserted inside of a round function of a cipher, and then the combinational path is balanced accordingly.

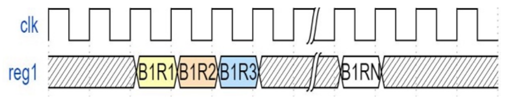
The basic iterative architecture, shown in Fig. 3 (left), is implemented first, and its maximum clock frequency, area and critical path are determined. Based on this information, we insert a pipeline register to reduce the critical path. The location of the pipeline register is chosen in such a way that the critical path between two adjacent registers is reduced and balanced. In this paper, we have implemented a two-stage inner-round pipelining, as shown in Fig. 3 on the right.

As shown in Fig. 2a and Fig. 3 on the left, in the basic iterative architecture [12-15], a single block of data is processed through N rounds in N clock cycles, and then the result is sent to the output. In the two-stage inner-round pipelined architecture, two blocks of data are read in two consecutive clock cycles, and the output is released after 2N+1 clock cycles. Additionally, two consecutive pairs of input blocks can be processed every 2N clock cycles.

A. *How does pipelining help?*

In the basic iterative architecture, shown in Fig. 3 on the left, the minimum clock period is given by the sum of the delay due to the register REG1 (d_{reg}), delay due to the multiplexer (d_{MUX}), delay due to the round (d_R), which is completely

a.



b.

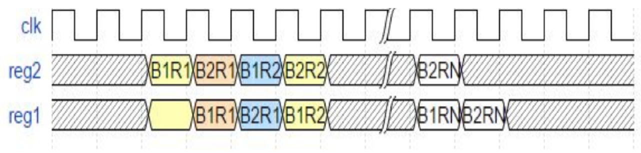


Figure 2. Timing Diagram: a. Basic Iterative Architecture and b. Inner-Round Pipelined Architecture with two pipeline stages.

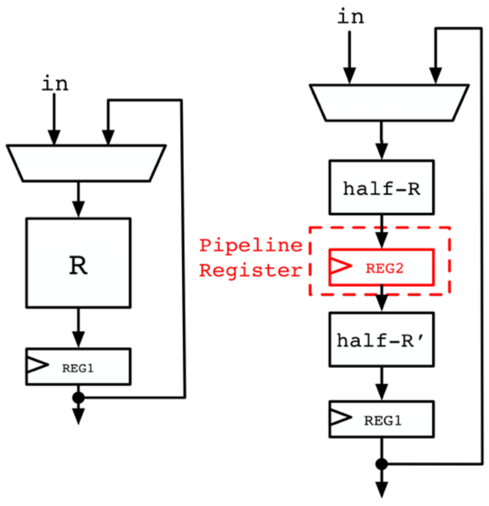


Figure 3. Generic Block Diagrams of the Basic Iterative Architecture (left) and the Inner-Round Pipelined Architecture with two pipeline stages (right) [12-14]

combinational logic, and the setup time (t_{setup}) of the register REG1. Then, the formula for the minimum clock period and maximum throughput of the basic iterative architecture is given by equations (1) and (2):

$$T_{clk} = d_{reg} + d_{MUX} + d_R + t_{setup} \tag{1}$$

$$\text{Throughput}_{Basic} = \text{block\_size}/(N \cdot T_{clk}) \tag{2}$$

The same formulas for the two-stage inner-round pipelined architecture, shown in Fig. 3 on the right, are given by equations (3) and (4):

$$T_{clk}' = d_{reg} + d_{MUX} + d_{half\text{-}R} + t_{setup} \tag{3}$$

$$\text{Throughput}_{Pipelined} = 2 \cdot \text{block\_size}/(2 \cdot N \cdot T'_{clk}) = \\ = \text{block\_size}/(N \cdot T'_{clk}) \tag{4}$$

where $d_{half\text{-}R}$ is a delay of a top half round, shown in Fig. 3 as half-R. We assume that $d_{half\text{-}R'} \leq d_{half\text{-}R} + d_{MUX}$.
From here,

$$\text{Throughput}_{Pipelined}/ \text{Throughput}_{Basic} = T_{clk}/ T'_{clk} \tag{5}$$

In the ideal pipelined architecture, the combinational logic of the round function is divided into perfect halves, so $d_{half\text{-}R}=0.5 \cdot d_R$.
Additionally, for the majority of ciphers, $d_{half\text{-}R}$ is much greater than $d_{reg} + d_{MUX} + t_{setup}$. From here,

$$\text{Throughput}_{Pipelined}/\text{Throughput}_{Basic} = T_{clk}/ T'_{clk} \approx d_R/d_{half\text{-}R} = 2 \tag{6}$$

At the same time, the equation (7) always holds:

$$\text{Throughput}_{Pipelined}/\text{Throughput}_{Basic} < 2 \tag{7}$$

However, increasing throughput by a factor of two is very challenging to achieve. Let us take an example in which the round is divided into two parts half-R and half-R', with 60% and 40% of the round function delay, respectively, i.e.,

$$T_{clk}' = d_{reg} + d_{MUX} + 0.6 \cdot d_R + t_{setup} \approx 0.6 \cdot d_R \tag{8}$$

In this case:

$$\text{Throughput}_{Pipelined}/\text{Throughput}_{Basic} \approx 1.67 \tag{9}$$

Thus, even a relatively small imbalance in the delays of both half-rounds (half-R and half-R') has a large detrimental effect on the improvement in speed.

B. *Why only two stages of pipelining?*

The number of stages in our method was limited to two because of the projected trade-off between the increased complexity of the design and the limited throughput and throughput/area ratio gain obtained by using more than two pipeline stages. Additionally, the limitations imposed by the the CAESAR Hardware API [17] and the GMU Development Package [25, 26] were taken into account as well.

III. Previous Work

The concept of inner-round, outer-round, and mixed pipelining was formalized in [12-14], with application to secret-key block ciphers, including Triple DES, and five final AES candidates.

Outer-round pipelining, used commonly before, assumed introducing pipeline registers only between full rounds of a partially or fully unrolled cipher. Although this kind of pipelining is very efficient in increasing circuit throughput, it rarely allows for any improvement in the throughput/area ratio.

On the other hand, inner-round pipelining, if possible and practical, may permit substantially increasing circuit frequency and throughput, at the cost of only minor increase in the circuit area. This feature is particularly true in case of FPGAs, in which area is measured in LUTs and Confgurable Logic Block (CLB) Slices (for Xilinx FPGAs) or ALUTs and Adaptive Logic Modules (ALMs) for Altera FPGAs. Since all LUTs/ALUTs are accompanied by the corresponding flip-flops (whether these flip-flops are used or not used), the pipeline registers, under some circumstences, may come virtually for free or at a relatively low cost associated with the more complex control logic.

For the inner round pipelining, the throughput is directly proportional to the maximum clock frequency. The following factors may limit the maximum clock frequency in this architecture: 1. delay of a single round divided by the number of pipeline stages, k, 2. delay of the longest indivisible operation, 3. delay of the control unit, 4. limit on the maximum latency, 5. limit on the maximum input/output bandwidth.

In [13], the pipeline register placement was solely dependent upon the critical path between any two adjacent registers, which was aimed to include only one level of CLBs. There was not any specific number of pipeline stages that was decided to be used in advance, as that could lead to irregular design, with pipelined registers inserted into the cipher elementary operations.

In Fig. 4, the improvement in clock frequency, and thus also the circuit throughput, is summarized for Triple DES and four final-round AES candidates. The ratio between the throughputs for the inner-round pipelined architecture and basic iterative architecture varied between 2.22 for Triple DES (3DES) up to 8.81 for Serpent. The obtained speed-up is inversely proportional to the frequency of the basic iterative architecture.

Additionally, Fig. 4 demonstrates that inner-round pipelining by itself allows accomplishing the highest or close to highest possible frequency, without the need of mixing it with outer-round pipelining (which leads to results shown in Fig. 4 using the mixed architecture bars).

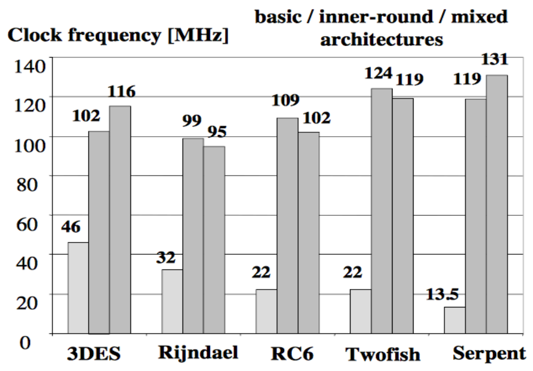


Figure 4. Results of implementing 3DES and four final AES candidates in Xilinx Virtex FPGAs. Maximum clock frequency of each of the three implemented architectures: basic iterative architecture, with inner-round pipelining, and with full mixed inner- and outer-round pipelining [13].

In [22], IDEA secret-key block cipher was implemented using inner-round pipelining, with 16 pipeline stages per round. This pipelining increased the maximum clock frequency relatively moderately, from 87.3 MHz for a non-pipelined combinational round, to 105.9 MHz for its deeply pipelined version. This relatively small improvement, by a factor of 1.21 can be attributed to a relatively simple round of the IDEA block cipher.

In [23], a high-speed hybrid implementation of the Grøstl hash function and the AES secret-key cipher was reported. The Grøstl P and Q transformations and the AES encryption round function E were sped up by a. processing P, Q and E in parallel, b. Using sub-round (inner-round) pipelining. In the latter method, a full round combinational logic was split into three balanced pipeline stages, which gave an increase in the maximum clock frequency by a factor of 1.4.

In [24], to support the high-speed Ethernet standard IEEE 802.3ba, the authors decided to exploit the parallelization and pipelining in the current authenticated cipher standard, AES-GCM, with the goal of achieving the 100 Gbps throughput rate. In this paper, AES with 14 outer-round pipeline stages was adopted, and to balance the design the 128-bit Galois Field multiplier was pipelined using four inner-round pipeline stages. With the use of four AES cores and four GF(2^{128}) multipliers and the final area optimization, the design achieved a frequency of 233 MHz, and the throughput in excess of 100 Gbit/s using Xilinx Virtex-5 FPGAs.

In this paper, inner-round pipelining is applied for the first time to CAESAR candidates. Taking into account the complexity of the datapaths and controllers required to implement these algorithms, the number of pipeline stages has been fixed at two, limiting the maximum speed-up that could be achieved, but making the redesign of the controllers and search for the optimal positions of pipelined registers in the corresponding datapaths much more practical.

IV. Development method

All seven pipelined implementations pursued in this study have been developed using as a starting point the open-source high-speed implementations of the respective CAESAR candidates, by E. Homsirikamol, W. Diehl [18, 27], F. Farahmand, and K. Minematsu, available at [21, 28]. These implementations were all based on the basic iterative architecture, and had their authorship and results reported in [5]. The implementation of the basic iterative architecture consists of the Datapath and Controller. The Datapath includes the round function and support for all other arithmetic and logic operations required for authenticated encryption/decryption. The Controller is a finite state machine, responsible for generating control signals for the Datapath.

Our pipelined designs are fully compliant with the CAESAR Hardware API [17], and take advantage of the GMU CAESAR Development Package [25] and the related Implementer's Guide [26].

The conversion from the basic iterative architecture to the two-stage pipelined architecture is shown schematically in Fig. 5. The Pipelined Design step involves adding pipeline registers in the Datapath and modifying the Controller accordingly. The

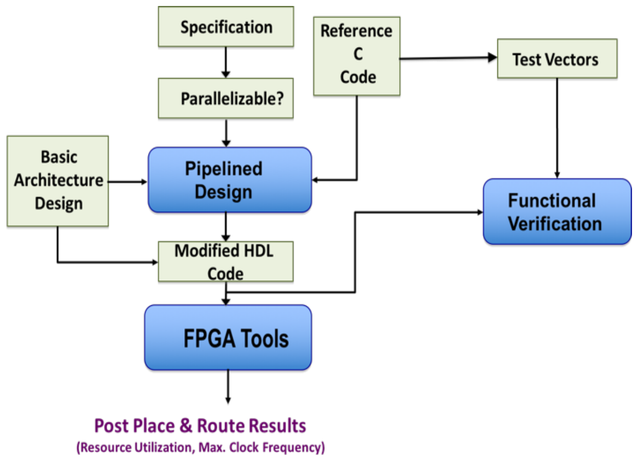


Figure 5. Development Methodology.

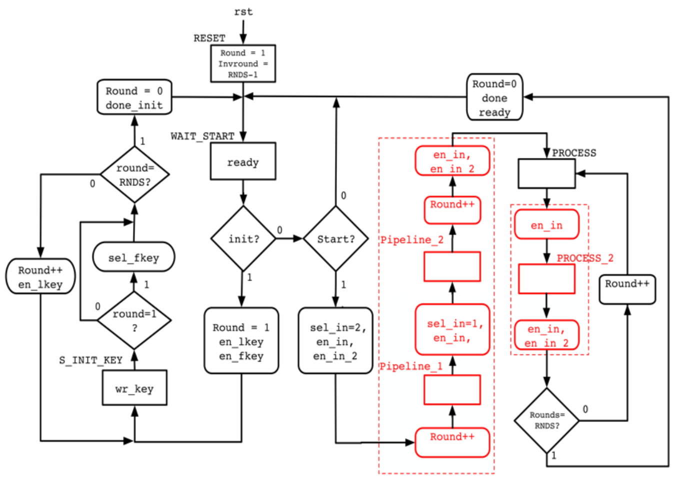


Figure 7. Algorithmic State Machine (ASM) chart of the AES Controller after the modifications required for two-stage inner-round pipelining.

modified control unit must generate all control signals necessary for processing of two blocks of data simultaneously. Some additional Datapath modifications may involve bus width changes, adding round constant calculations for a second block of data, adding registers to buffer the data, etc. The Controller modifications involve adding extra states and support for additional control signals (e.g., the enable signal of the pipelined register, the select signals of multiplexers, etc.).

After all necessary modifications, the pipelined implementation is functionally verified using the corresponding reference C code as a source of test vectors. The maximum clock frequency, throughput, and resource utilization are determined and compared with the values for the basic iterative architecture. In case the speed-up is insufficient, a different location of pipeline registers is attempted.

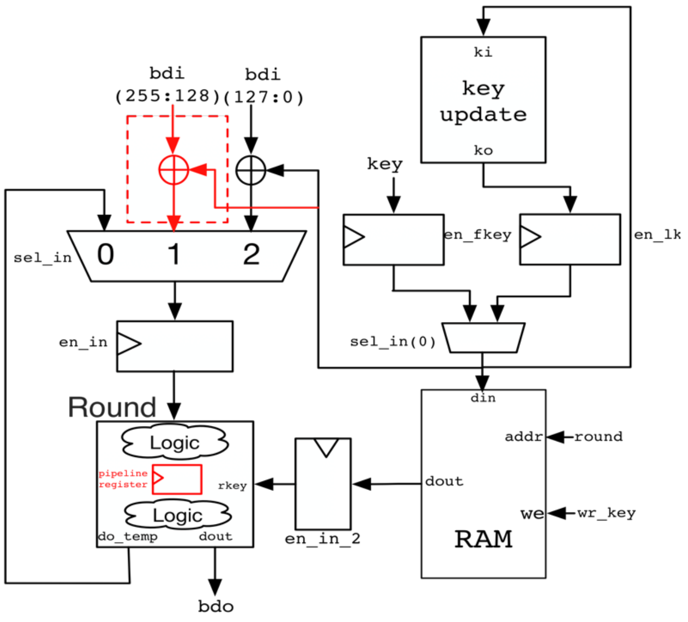


Figure 6. Block diagram of the AES Datapath after the modifications required for two-stage inner-round pipelining.

In Figs. 6 and 7, we provide an example of the conversion for a simply case of a secret-key block cipher AES. Round keys are assumed to be precomputed and stored in RAM. The parts of both figures surrounded with red dashed boxes are the extensions/modifications required for the conversion of the basic iterative architecture to the two-stage pipelined architecture. Two inputs, carried by the two halves of the input signal bdi(255:0) are provided to the AES unit in two consecutive clock cycles. The corresponding outputs are generated RNDS clock cycles later, concatenated, and sent to the output bdo.

V. SUMMARY OF PIPELINED IMPLEMENTATIONS

From Round 2 and Round 3 of the CAESAR competition, seven parallelizable candidates were chosen, for which the maximum clock frequency was the lowest in the basic iterative architecture design [5]. Basic parameters of selected candidates are summarized in Table 1.

TABLE I. BASIC PARAMETERS OF ALL INVESTIGATED CANDIDATES

Candidate Name	**Key Size** (bits)	**Block Size** (bits)	**Tag Size** (bits)	**Rounds**
SCREAM	128	128	128	10
AES-COPA	128	128	128	10
COLM	128	128	128	10
AES-OTR	128	128	128	10
MINALPHER	256	256	128	17
DEOXYS	128	128	128	14
OCB	128	128	128	10

A. SCREAM

The Round function of SCREAM [6] is built using the combination of S-boxes and L-boxes [1] [2], as shown in Fig. 8. As discussed in the development method, first, we get the initial estimates of maximum clock frequency and throughput. Then, by inserting a register in the round function, the critical path is minimized. The register is added at different locations and each location checked for the shortest critical path. The pipelined register has been introduced in the encryption and decryption paths, after S_box and Inv_L_box, respectively, as shown in Fig. 8. Round function is then adjusted into a form, where the circuit can process two blocks of data in parallel. Then, respective changes are made in the controller to support processing of two blocks of data in parallel. Using this approach, the critical path delay was reduced from 10.8 ns to 5.8 ns. Thus, an 84% increase in maximum clock frequency was achieved.

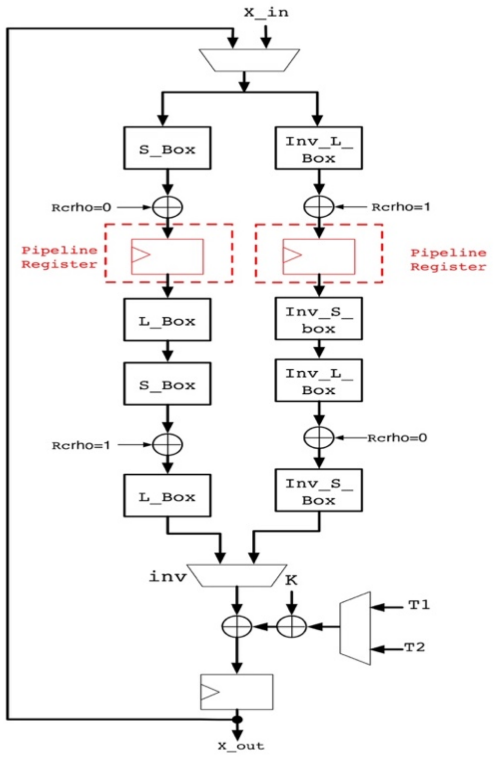


Figure 8. SCREAM: Pipelined Round.

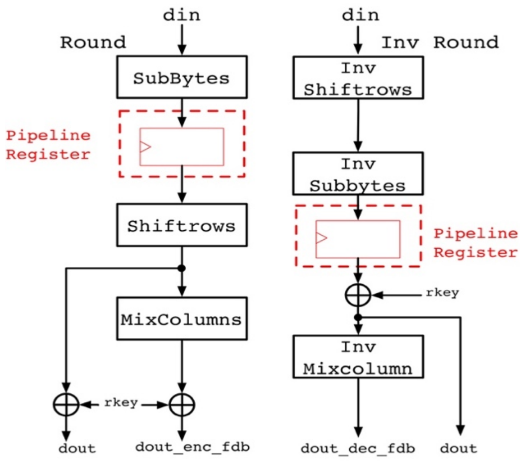


Figure 9. AES-COPA: Pipelined Round.

B. AES-COPA

The Round function of AES-COPA [10] consists of the AES round as the basic building block, as shown in Fig. 9. The pipeline register was added at all the available locations in the round function and verified for the functional correctness and critical path delay. For example, in the Round from Fig. 9, the register was added after SubBytes and then the design was checked for the critical path delay. Afterward, the position of the register was moved after the Shiftrows and the same process was repeated until the smallest value of critical path delay was found. Using this approach, the critical path delay was reduced from 8.3 ns to 4.76 ns. Thus, an increase of 75% in maximum clock frequency and throughput was achieved.

C. COLM

COLM [20] is derived from two CAESAR candidates AES-COPA and ELmD. The Round function of COLM consists of the same components as AES round, and the basic building blocks are shown in Fig. 10. Similar to other designs the pipelined register was added to COLM, but this did not really help in reducing the critical path delay. The figure just shows the logic inside the round function, approximately half of the critical path was located outside of the round function, so the pipelined register was added at the beginning of the round function to divide the critical path into two halves. Using this approach, the critical path delay was reduced from 8.92 ns to 5.55 ns. Thus, a 60% improvement in maximum clock frequency and throughput was achieved after inner-round pipelining.

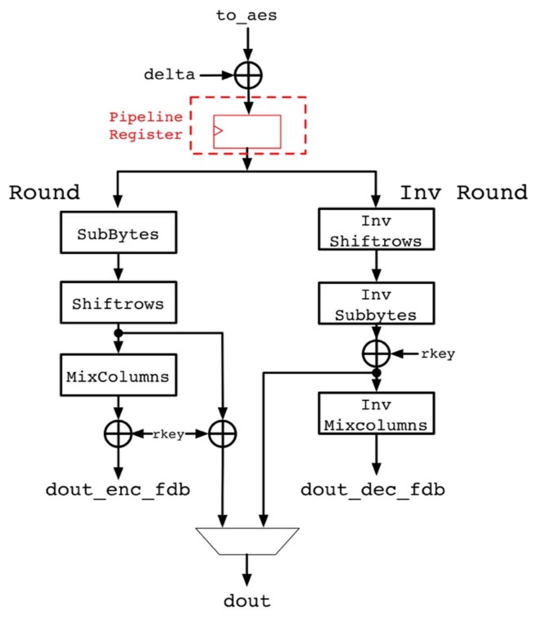


Figure 10. COLM: Pipelined Round.

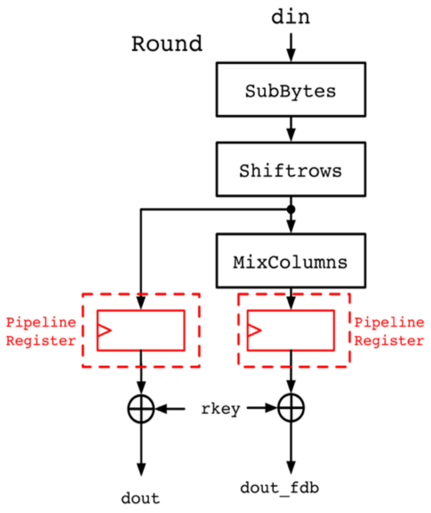


Figure 11. AES-OTR Pipelined Round.

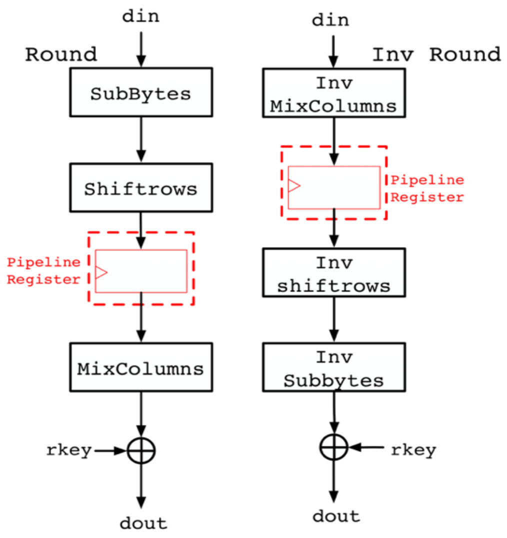


Figure 12. DEOXYS: Pipelined Round.

D. AES-OTR

The round function of AES-OTR [9] consists of the same components as AES, as shown in Fig. 11. Similar method of placing pipeline register at different locations and checking for the smallest critical path delay was applied. As a result, the critical path delay was reduced from 6.66 ns to 4.25 ns. Thus, a 56% improvement in maximum clock frequency and throughput was achieved after inner-round pipelining.

E. DEOXYS

The Round function of Deoxys [19] consists of the same building blocks as AES, as shown in Fig. 12. Pipelined implementation for Deoxys was straightforward. Just following the methodology in Section IV gave good results. The critical path delay was reduced from 5.15 ns to 3.69 ns. Therefore, an increase by 39% in maximum clock frequency and throughput was achieved after inner-round pipelining.

F. MINALPHER

The Round function of Minalpher consists of Subnibbles, Shuffle Rows, Swap Matrix, Mix Column and Add Round Constant [3] [4], with all these operations described in [11]. The round function of Minalpher had many potential locations for the pipeline registers.

Initially two 128 bit registers were added after Swap Matrix and the circuit tested to determine the critical path reduction. The reduction corresponding to this placement was very low, as the critical path kept shifting from the round function to the tweak calculator. The same was the case with almost all the locations. The location of the pipeline register shown in Fig. 13 gave the best results. The critical path delay was reduced from 5.95 ns to 4.50 ns. Thus, a 32% increase in maximum clock frequency and throughput was achieved after inner-round pipelining.

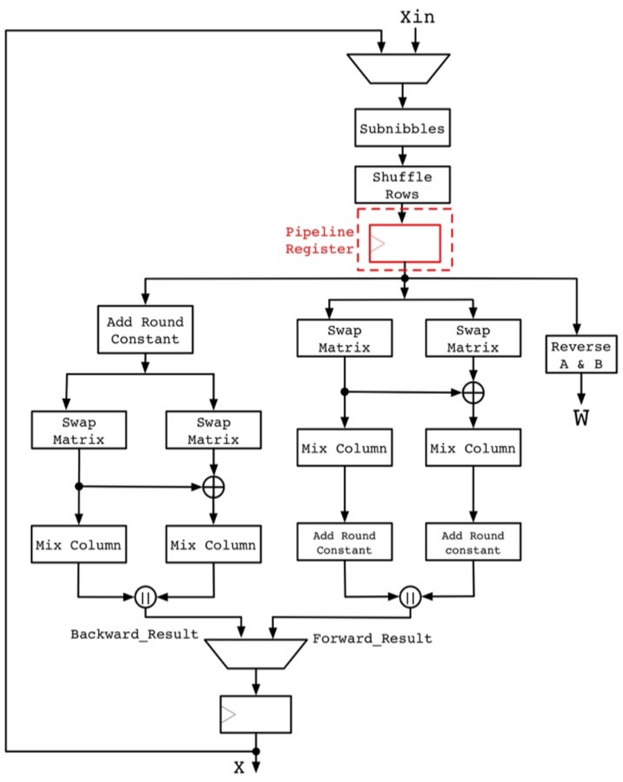


Figure 13. MINALPHER: Pipeline Round.

G. OCB

The Round function of OCB [8] consists of AES as a basic building block. The forward round of OCB was a similar case to COLM, namely, half of the critical path was outside of the round function. Thus, after adding the pipeline register at different locations, there was not much reduction in the critical path delay. Instead, the critical path kept shifting to the key scheduling block of the inverse round. So, a register at the end

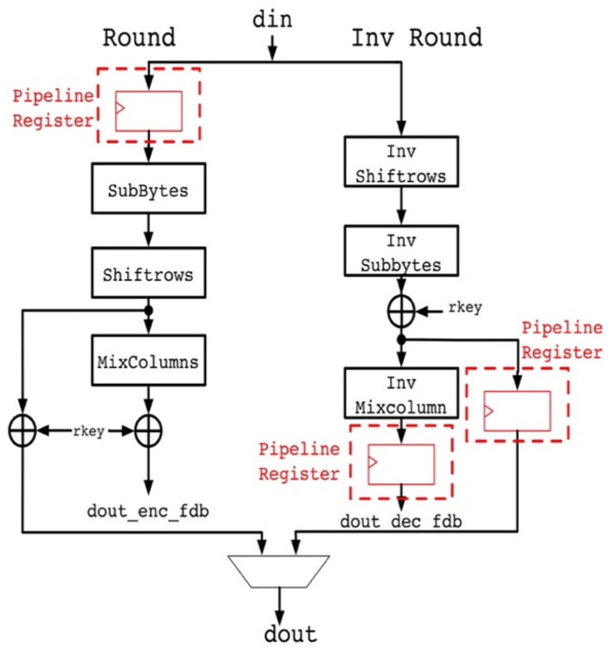


Figure 14. OCB: Pipelined Round.

of inverse round was added. The register placement, shown in Fig. 14, gave the optimum performance. The critical path delay was reduced from 5.81 ns to 4.52 ns. Thus, a 28% improvement in maximum clock frequency and throughput was achieved after inner-round pipelining.

VI. Performance Evaluation

Performance of each candidate was evaluated based on the improvement in the maximum clock frequency and throughput, as well as penalty in terms of the circuit area. All of the following results were generated using the largest FPGA available in the student version of Xilinx ISE v14.7, namely, Xilinx Virtex 6 XC6VLX75T-3FF784.

In Table II, we summarize the numbers of clock cycles per block and the throughput formulas for all investigated candidates. The corresponding information for the basic iterative architecture was obtained from [5] and [21]. The number of clock cycles per block for our pipelined architectures was derived by analysis of our designs, and verified using functional simulation.

Based on Table III and Fig. 15, it can be observed that the lower the value of the maximum clock frequency in the basic iterative architecture the higher the frequency gain in the pipelined architecture. The same relation applies also to the throughput as well.

The increase in the area from basic iterative architecture to the pipelined architecture is shown in Table IV. This increase is calculated based on both the number of LUTs and Slices. The increase in the number of LUTs ranged from 9% for SCREAM to 47% for AES-OTR. The increase in the number of Slices was significantly larger. It varied from 49% for Minalpher to 100% for AES-COPA.

TABLE II. Number of Clock Cycles per Block and Throughput Formula for Each Candidate

Candidate	**Number of clock cycles per block**		**Throughput Formula** (Mbits/sec)
	Basic Arch	**Pipelined Arch**	
SCREAM	11	22	11.63* f_{clk}
AES-COPA	11	22	11.63* f_{clk}
COLM	11	22	11.63* f_{clk}
AES-OTR	12	24	10.66* f_{clk}
DEOXYS	15	30	8.53* f_{clk}
MINALPHER	19	38	13.47* f_{clk}
OCB	12	24	10.66* f_{clk}

TABLE III. Maximum Clock Frequency and Throughput Comparison Between Basic and Pipelined Architecture

Candidate	**Maximum Clock Frequency** (MHz)		**Throughput** (Mbits/sec)	
	Basic Arch	**Pipelined Arch**	**Basic Arch**	**Pipelined Arch**
SCREAM	92	170	1071	1977
AES-COPA	120	210	1396	2442
COLM	112	180	1303	2093
AES-OTR	150	235	1600	2507
DEOXYS	194	271	1655	2311
MINALPHER	168	222	2264	2991
OCB	172	221	1835	2357

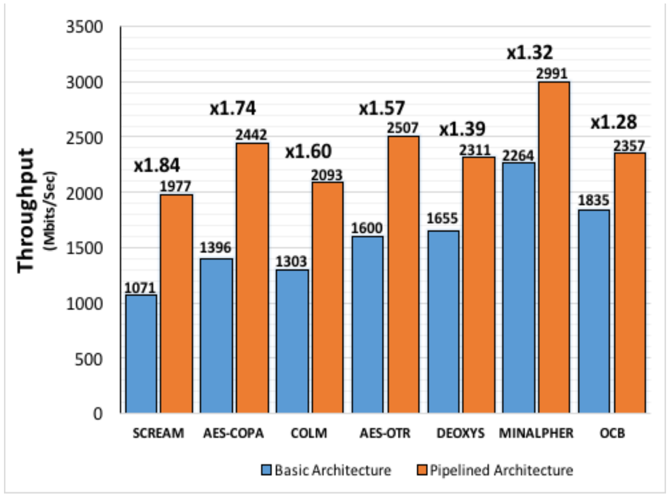


Figure 15. Throughput: Basic Architecture vs. Pipelined Architecture.

From Table V and Fig. 16, it can be observed that the increase in the Throughput to Area ratio from the basic iterative architecture to the inner-round pipelined architecture was the highest for SCREAM. In Minalpher the ratio was reduced, as there was only 32% increase in maximum clock frequency and 44% increase in area. An improvement was observed for all other candidates.

TABLE IV. AREA COMPARISON BETWEEN BASIC AND PIPELINED ARCHITECTURE

Candidate	Area (LUTs)		Area (Slices)	
	Basic Arch	Pipelined Arch	Basic Arch	Pipelined Arch
SCREAM	3644	3968	1546	2442
AES-COPA	4902	6484	2216	4431
COLM	6754	8337	2447	3962
AES-OTR	5058	7443	2219	3637
DEOXYS	2825	3943	1107	1805
MINALPHER	7836	11285	3974	5915
OCB	3312	3673	1742	2905

TABLE V. THROUGHPUT TO AREA RATIO COMPARISON BETWEEN BASIC AND PIPELINED ARCHITECTURE

Candidate	Throughput/ Area (Mbits/(sec*LUTs))		Throughput/ Area (Mbits/(sec*Slices))	
	Basic Arch	Pipelined Arch	Basic Arch	Pipelined Arch
SCREAM	0.29	0.50	0.69	0.81
AES-COPA	0.28	0.38	0.63	0.55
COLM	0.19	0.25	0.53	0.53
AES-OTR	0.32	0.34	0.72	0.69
DEOXYS	0.59	0.60	1.50	1.28
MINALPHER	0.29	0.27	0.57	0.51
OCB	0.55	0.64	1.05	0.81

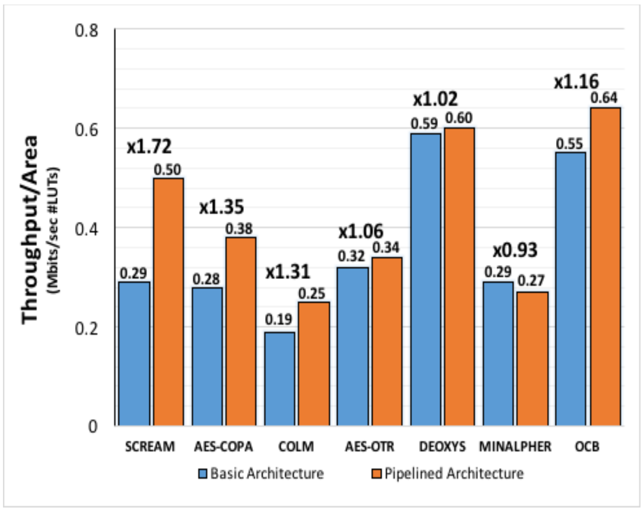


Figure 16. Throughput to Area Ratio: Basic Architecture vs. Pipelined Architecture, assuming that area is expressed using LUTs.

VII. CONCLUSIONS

The improvement in the maximum clock frequency and throughput depends on the algorithm and its critical path. Based on the results presented in Section VI, the performance gain is inversely proportional to the frequency and throughput in the basic iterative architecture. Our results have demonstrated the improvement in the clock frequency and throughput by a factor varying from x1.28 for OCB to x1.84 for SCREAM, and the improvement in the throughput to area ratio (with area expressed using LUTs) by a factor varying from x0.93 for Minalpher to x1.72 for SCREAM. Improvement in Throughput/#LUTs was observed in six candidates, all except Minalpher. In terms of the relative performance of pipelined implementations, the top four candidates were SCREAM, AES-COPA, COLM and AES-OTR, all with the frequency and throughput gains exceeding 50%.

ACKNOWLEDGMENT

This work has been partially supported by NSF Grant #1314540.

REFERENCES

[1] M. Liskov, R. L. Rivest, and D. Wagner, Tweakable block ciphers, Journal of Cryptology, vol. 24, no. 3, 2011, pp. 588-613.
[2] S. Even and Y. Mansour, A Construction of a Cipher from a Single Pseudorandom Permutation, Journal of Cryptology, vol. 10, no. 3, Jun. 1997, pp. 151-162.
[3] K. Kurosawa, Power of a public random permutation and its application to authenticated-encryption. IACR Cryptology ePrint Archive, 2002:127.
[4] K. Kurosawa. Power of a public random permutation and its application to authenticated encryption. IEEE Transactions on Information Theory, vol. 56, no. 10, 2010, pp. 5366-5374.
[5] Cryptographic Engineering Research Group (CERG). (2017) Database of FPGA Results for Authenticated Ciphers. [Online]. Available: https://cryptography.gmu.edu/athenadb/fpga_auth_cipher/rankings_view
[6] V. Grosso, G. Leurent, F. Standaert, K. Varici, F. Durvaux, L. Gaspar, and S. Kerckhof. SCREAM and iSCREAM. Submission to CAESAR, March 2014, see [7].
[7] "CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness," available at http://competitions.cr.yp.to/caesar.html
[8] T. Krovetz and P. Rogaway. The OCB Authenticated-Encryption Algorithm. Submission to CAESAR, May 2014, see [7].
[9] K. Minematsu. AES-OTR v3.1. Submission to CAESAR, Sep. 2016, see [7].
[10] E. Andreeva, A. Bogdanov, A. Luykx, B. Mennink, E. Tischhauser, and K. Yasuda. AES-COPA v.2. Submission to CAESAR, see [7].
[11] Y. Sasaki, Y. Todo, K. Aoki, Y. Naito, T. Sugawara, Y. Murakami, M. Matsui, S. Hirose. Minalpher v1.1. Submission to CAESAR, Aug. 2015, see [7].
[12] K. Gaj and P. Chodowiec, "Comparison of the Hardware Performance of the AES Candidates Using Reconfigurable Hardware," Proc. 3rd Advanced Encryption Standard Conference, New York, Apr. 2000, pp. 40-54.
[13] P. Chodowiec, P. Khuon, and K. Gaj, "Fast Implementations of Secret-Key Block Ciphers Using Mixed Inner- and Outer-Round Pipelining," ACM/SIGDA Ninth International Symposium on Field Programmable Gate Arrays, Monterey, CA, Feb. 2001, pp. 94-102.
[14] K. Gaj and P. Chodowiec, "Fast Implementation and Fair Comparison of the Final Candidates for Advanced Encryption Standard using Field Programmable Gate Arrays," LNCS 2020, Progress in Cryptology - CT-RSA 2001, Ed. D. Naccache, RSA Conference 2001 - Cryptographers' Track, San Francisco, Apr. 2001, pp. 84-99.
[15] E. Homsirikamol, M. Rogawski, and K. Gaj, "Throughput vs. Area Trade-offs in High-Speed Architectures of Five Round 3 SHA-3 Candidates Implemented Using Xilinx and Altera FPGAs," in LNCS 6917, Cryptographic Hardware and Embedded Systems - CHES 2011, Nara, Japan, Sep. 28-Oct. 1, pp. 491-506.
[16] E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, M.U. Sharif, and K. Gaj, "A Universal Hardware API for Authenticated Ciphers," 2015 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2015, Mayan Riviera, Mexico, Dec. 7-9, 2015.
[17] E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, P. Yalla, J.P. Kaps, K. Gaj, "CAESAR Hardware API," Cryptology ePrint Archive, Report 2016/626, Internet: http://eprint.iacr.org/2016/626.pdf [June. 30, 2017].

[18] W. Diehl and K. Gaj, "RTL Implementations and FPGA Benchmarking of Three Authenticated Ciphers Competing in CAESAR Round Two," 19th Euromicro Conference on Digital System Design - DSD 2016, Limassol, Cyprus, Aug. 31-Sep. 2, 2016.
[19] J. Jean, I. Nikolic, T. Pyrin, Y. Seurin. Deoxys v1.41. Submission to CAESAR, Oct. 2016, see [7].
[20] E. Andreeva, A. Bogdanov, N. Datta, A. Luykx, B. Mennink, M. Nandi, E. Tischhauser, K. Yasuda. COLM v1. Submission to CAESAR, Sep. 2016, see [7].
[21] GMU Source Code of Round 2 & Round 3 CAESAR Candidates, AES-GCM, AES, AES-HLS, and Keccak Permutation F, available at https://cryptography.gmu.edu/athena/index.php?id=CAESAR_source_codes
[22] A. Hamalainen, M. Tommiska, J. Skytta, "6.78 Gigabits per Scecond Implementation of the IDEA Cryptographic Algorithm," 12th Conference on Field-Programmable Logic and Applications, FPL 2002, Montpellier, France, Sep. 2002, pp. 760-769.
[23] K. Guo and H. M. Heys, "A Pipelined Implementation of the Grøstl Hash Algorithm and the Advanced Encryption Standard," 26th Annual IEEE Canadian Conference on Electrical and Computer Engineering, CCECE 2013, Regina, SK, 2013, pp. 1-4.
[24] L. Henzen and W. Fichtner, "FPGA Parallel-Pipelined AES-GCM Core for 100G Ethernet Applications," 36th European Solid State Circuits Conference, ESSCIRC 2010, Sevilla, Spain, pp. 202-205.
[25] E. Homsirikamol, P. Yalla, F. Farahmand, J.-P. Kaps, and K. Gaj, "Development Package for Hardware Implementations Compliant with the CAESAR Hardware API," available at https://cryptography.gmu.edu/athena/index.php?id=CAESAR
[26] E. Homsirikamol, P. Yalla, F. Farahmand, J.-P. Kaps, and K. Gaj "Implementer's Guide to Hardware Implementations Compliant with the CAESAR Hardware API", available at https://cryptography.gmu.edu/athena/index.php?id=CAESAR
[27] W. Diehl and K. Gaj, RTL Implementations and FPGA Benchmarking of Selected CAESAR Round Two Authenticated Ciphers, Microprocessors and Microsystems, vol. 52, July 2017, pp. 202-218.
[28] VHDL/Verilog Code of Round 2 CAESAR Candidates: Summary I, available at https://cryptography.gmu.edu/athena/CAESAR_HW_Summary_1.html

Appendix A

TABLE A1. Capability for Parallel Processing of the Associated Data, Message, and Ciphertext Blocks for Round 3 CAESAR Candidates, Determined Based on Their Respective Specifications

Candidate	Associated Data	Message	Ciphertext
ACORN	X	X	X
AEGIS	X	X	X
AES-OTR (Parallel ADP)	✓	✓	✓
AES-OTR (Serial ADP)	X	✓	✓
AEZ	✓	✓	✓
Ascon	X	X	X
CLOC	X	X	✓
SILC	X	X	✓
COLM	✓	✓	✓
Deoxys	✓	✓	✓
JAMBU	X	X	X
Ketje	X	X	X
Keyak	X	X	X
MORUS	X	X	X
NORX	X	X	X
OCB	✓	✓	✓
Tiaoxin	X	X	X

TABLE A2. Capability for Parallel Processing of the Associated Data, Message, and Ciphertext Blocks for the CAESAR Candidates which were Eliminated after Round 2, Determined Based on Their Respective Specifications

Candidate	Associated Data	Message	Ciphertext
HS1-SIV	✓	✓	✓
ICEPOLE	X	X	X
Joltik	✓	✓	✓
Minalpher	✓	✓	✓
OMD	✓	X	X
PAEQ	✓	✓	✓
POET	✓	✓	✓
PRIMATEs APE	X	X	X
PRIMATEs HANUMAN	X	X	X
PRIMATEs GIBBON	X	X	X
SCREAM	✓	✓	✓
SHELL	X	✓	✓
STRIBOB	X	X	X
TriviA-ck	X	X	X