Energy-Efficient and High-throughput Implementations of Lightweight Block Cipher

Pulkit Singh, Piyush Modi, Bibhudendra Acharya, Rahul Kumar Chaurasiya

Abstract*: Security in resource-constrained devices has drawn the great attentions to researchers in recent years. To make secure transmission of critical information in such devices, lightweight cryptography algorithms come in light to large extend. KLEIN has been popular lightweight block cipher used to overcome such issues. In this paper, different architectures of KLEIN block cipher are presented. One of designs enhances the efficiency with regard to the throughput at the expense of a larger area. In order to make such designs, the pipelined registers are placed on different positions in datapath algorithm. The proposed design transforms the data input to protected output with the speed of 2414.13 Mbps for xc5vlx50t-3ff1136 device. In addition, the second design implementation completes either one or more than one round in only one clock and gives energy-efficient and high throughput implementations. Due to this, a trade-off between area and speed can be analyzed for high-speed applications. Moreover, this proposed design shows that with increasing the area of cipher implementation results in more transformation of plaintext into ciphertext. All results are verified and simulated for various families of Xilinx ISE design suite.

***Keywords*: FPGA, KLEIN cipher, Lightweight algorithm, Throughput, Resource-constrained environments, Security, Symmetric encryption.**

I. INTRODUCTION

Information is being transmitted through an unsecure channel. It is desired for researchers keeping information secure with constrained resources. In order to provide security to such devices, conventional cryptography methods like Data Encryption Standard (DES) and Advanced Encryption Standard (AES) cannot be utilized [1], [2]. Due to this, lightweight cryptography algorithms draw a lot of attention recently. These algorithms play a vital role to secure resource constrained devices. Moreover, resource limited environments have constrained commodity in terms of area, power, and memory.

Lightweight cryptographic algorithms have four basic primitives [3]. Out of these, a block cipher has been mostly utilized cryptographic primitives. This primitive has used widely in information encryption, random bit generation, message authentication, message hashing, etc. Apparently, the most generally utilized block cipher on the planet is AES, which has been built up as different worldwide guidelines. AES indicates great execution figures on most programming and equipment stages and is commonly viewed as secure in the wake of making due around 15 years of exhaustive cryptanalysis. However, a few shortcomings have been initiated, since many block ciphers including AES, have been utilized as hardware implementation for lightweight resource-constrained applications. These implementations are called as ISO lightweight cryptography (ISO/IEC 29192-II) standard [4]. The fundamental feature of the lightweight block ciphers is the effective hardware implementation with low asset. So as to accomplish that objective, the vast majority of them utilize straightforward structures with large number of rounds and small block sizes. Nonetheless, those structure approaches more often than not prompt low execution, and is a long way from our thought for programming encryption. Subsequently, we have planned another square figure giving a quick encryption on basic programming stages.

There are various lightweight algorithms such as PRESENT [5], HIGHT [6], PICCOLO [7], QTL [8], RECTANGLE [9], KLEIN [10], LILLIPUT [11], etc. These algorithms secure the systems which require limited resources for implementation in Field Programmable Gate Array (FPGA). Some ciphers provide security through software implementation whereas some are beneficial to develop security features in hardware implementation. Lightweight cipher hardware application is extremely suited for resource-constrained contexts. The bit-slice permutation is best suitable for developing software realization of a lightweight cipher than bit-wise implementation [12]. It shows better characteristics in terms of code size, Random Access Memory (RAM) size, and latency. Involutive characteristics are implemented in its components to execute the decryption module. Due to this approach, it saves the cost of decryption and lowers latency.

A. Related Work

Advanced Encryption Standard (AES) has drawn more attentions for other ciphers. Various Hardware implementations on FPGA have been developed with different techniques. Moreover, throughput can be enhanced

Revised Manuscript Received on December 12, 2019.

***** Correspondence Author

Pulkit Singh, Electronics & Communication Engineering department, National Institute of Technology Raipur, Raipur, India.
Email: pksingh.phd2015.elx@nitrr.ac.in

Piyush Modi, Electronics & Communication Engineering department, National Institute of Technology Raipur, Raipur, India.
Email: modipiyush16@gmail.com

Bibhudendra Acharya*, Electronics & Communication Engineering department, National Institute of Technology Raipur, Raipur, India.
Email: bacharya.etc@nitrr.ac.in

Rahul Kumar Chaurasiya, Electronics & Communication Engineering department, Malaviya National Institute of Technology Jaipur, Jaipur, India.
Email: rahul.ece@mnit.ac.in

by numerous methods like Partial rolling [13], Loop-unrolling [14], and an Iterative design [15] in hardware realization. Additionally, throughput and operating frequency can be improved by introducing pipelining and sub-pipelining strategies into block ciphers. In [16], [17], the authors utilized pipelined strategies in between the rounds and within the rounds for the AES. In addition, parallel pipelined architecture achieved a high-speed design with efficient hardware implementation [18]. A 4-stage pipelined S-box and 32-bits general purpose 5-stage pipelined processor are presented advanced Very Large-Scale Integration (VLSI) architectures [19], [20].

To optimize area and power, a loop-based approach is preferred. Authors in PRESENT cipher presented round-based architectures with a minimal hardware footprint of 1570 GE, whereas authors have achieved several security levels for AES 32-bits cipher with high-throughput, ultralow-power, ultralow-energy, and low-cost design by implementing various optimization strategies [21]. In addition, two optimized architectures for PRESENT cipher are implemented by compromising the low-resources results with security. They have designed two architectures with 16-bits datapath in each by varying the key sizes of 128-bits and 80-bits. PRESENT cipher is considered an ultra-lightweight block cipher among all lightweight ciphers having sufficient security features [22]. Therefore, most of authors have made changes in the hardware implementations of PRESENT cipher. Authors have brought two area-optimized architectures by demonstrating 16-bits datapath, where one of design has introduced parallel processing approach for high performance implementation [23]. In addition, a very compact hardware implementation can also be achieved by introducing SRL16 based implementations in the lightweight ciphers. This storing element stores and shifts the values when clock cycle triggers [24]. Moreover, parallel processing architecture shows a good throughput in the paper [25].

B. *Structure of the paper*

Two design approaches are developed in this paper. These designs are well suited to target resource constrained devices such that high throughput and less latency can be achieved. To accomplish these goals, the following architectures are contributed on this paper:

- The first strategies are demonstrated by keeping registers to different places. These are defined as full-pipeline, sub-pipeline and full-sub pipeline where pipelining registers are inserted at appropriate positions. This implementation enhances the data throughput of cipher.
- Also, next designs are planned showing the trade-off between area and speed. To complete such performance, the designs are implemented by completing more than one round in single clock cycles.

The rest of the paper is organized in the following way. Part II briefly defines the datapath algorithm and key scheduling for KLEIN cipher. In Section III, the proposed designs are elaborated the design methodology for hardware implementation. Section IV provides experimental result information. Section V discusses the simulation outcomes and compares them. Part VI concludes the work.

II. ALGORITHM OVERVIEW

A. *Datapath Processing*

KLEIN is a lightweight block cipher, which has been developed on legacy sensor platforms. KLEIN performs same round function as described by: Sub Nibbles, Rotate Nibbles, *MixNibbles*. KLEIN which focused mainly on Substitution-Permutation network works on 64-bit of plaintexts, and keys ranges between 64-bit, 80-bit and 96-bit which used for encrypt the plaintext to get ciphertext with 12, 16 and 20 rounds respectively. A non-linear procedure, Sub Nibbles which has been used to provide a unique security to resource constrained devices in KLEIN. Plaintext of 64-bits is XORed with 64-bits of the key for the 64-bits of key size, before performing round function in each round, whereas LSB 64-bits are extracted from the round keys for 80 and 96-bits of key scheduling. These 64-bits will be divided into 16 different nibbles that will be passed by S-boxes. To save the implementation costs of decryption process 4-bit S-box have been used in KLEIN cipher because it is a 4x4 involuting S-boxes.

This algorithm is processed based on byte-oriented basis. During the *RotateNibbles* operation, every two bytes is shifted left in each round. In *MixNibbles* step, mixing of columns is performed as in AES algorithm with the same matrix multiplication.

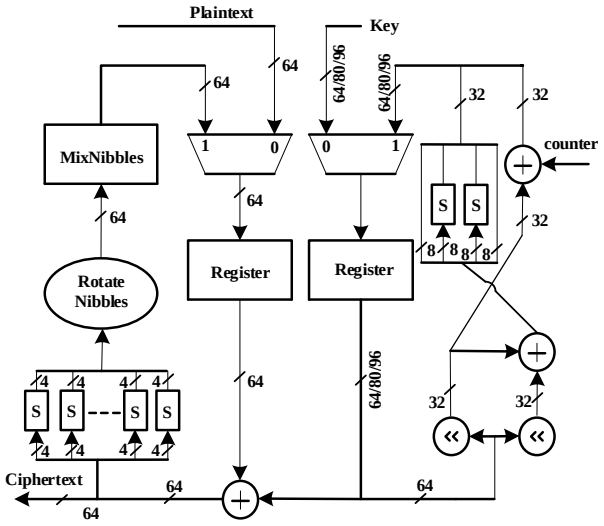

Fig. 1. Basic Hardware Implemention of KLEIN

B. *Key Scheduling*

Feistel-based structure is best suited for weak key stands hence we have used this for KLEIN key scheduling. To resist the slide attacks we have used an incremental round counter, which adds constants to each round to resist slide attacks. At the time when round transformation happens same time subkeys also calculated on the-fly mode which reserves memory requirement for between states. Whichever key we will be using in 64-bit, 80-bit and 96-bit structures always first 64 bits will be used for algorithm procedures. The key scheduling of

Retrieval Number: B10221292S19/2019©BEIESP
DOI: 10.35940/ijitee.B1022.1292S19

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

KLEIN for 64-bits is modelled as given below in equation (1).

$$K = k_{63}\ k_{62}\ k_{61}\ldots\ldots\ldots k_2\ k_1\ k_0$$
$$K_1 = k_{63}\ k_{62}\ k_{61}\ldots\ldots\ldots k_{34}\ k_{33}\ k_{32}$$
$$K_2 = k_{31}\ k_{30}\ k_{29}\ldots\ldots\ldots k_2\ k_1\ k_0$$
$$K_{11} = K_1 << 8\ \&\ K_{21} = K_2 << 8$$
$$K_{12} = K_{21}\ \&\ K_{22} = K_{11}\ \text{xor}\ K_{21} \quad (1)$$
$$K_f[44:40] = K_{12}[44:40]\ \text{xor}\ RC^i$$
$$K_f[11:08] = S(K_{22}[11:08])$$
$$K_f[15:12] = S(K_{22}[15:12])$$
$$K_f[19:16] = S(K_{22}[19:16])$$
$$K_f[23:20] = S(K_{22}[23:20])$$

The subscript of K is f which denotes final key for the described rounds, RCi have been used for round constants representation which is one incremented value for different rounds, S denotes S-box input values.

Datapath and Key Scheduling are the two part of the KLEIN by which this algorithm is described. Each round performs different operation hence after each round value at output of algorithm will be different and more confusing as rounds increases for all sizes of keys. 16 S-boxes and 4 S-boxes have been used for Datapath and Key Scheduling respectively. Key scheduling used in KLEIN uses Feistel structure which was mainly created for increasing the security of cipher. Moreover, for 80 and 96-bits key, MSB 64-bits are taken as the round key.

III. IMPLEMENTATION WORK

By taking various design strategies, two architectures are being developed that are advancing the design features. The first approach employs pipelined registers either at the start of each round or at between stages of each round. These rregisters are placed at their optimal placement, where critical path delay is large. The second proposed structure completes either one or more than one round in one clock cycle. Thus, based on the forementioned approaches, the following sub-section are examined two hardware implementations for KLEIN cipher with their detailed technical descriptions.

A. Pipelined Architectures

Datapath and key scheduling of KLEIN cipher are generated for the proposed architectures as explained in the overview section. The only difference is here that the pipelined registers are placed at different positions in the round function. For each implementation, data is transforming from one pipelined register to the next register when clock is changed. After a total of 12 clock cycles, the output starts to come out sequentially. Fig. 2 shows the pipelined design of KLEIN cipher by placing registers at the start of each round. Therefore, total of twelve 64-bits pipelined registers are used to encrypt the data. These registers are utilized for the temporary storage of data during the execution of around. The pipeline registers transport both the information and the control signal starting with one pipeline stage then onto the next. One stage of pipeline performs one operation of any instruction at once; therefore, each combinational operation is made on behalf of instructions taking place between any couples of pipeline registers. Pipelined registers transform the data from registers to next registers in one clock cycle. Since, there are twelve rounds for 64-bits key scheduling, which produce the ciphertext in total of twelve clock cycles. This process increases the maximum frequency resulting in high throughput.

Moreover, sub-pipelined architecture inserts register between stages at each round. Each stage takes some time for completing own operation. Therefore, these pipelined resisters reduce the critical path during the execution of each operation. Data are transforming from one register to the next register at every clock cycle. After a total of twenty four clock cycles, the output has started to come out continuously. Fig. 3 shows the sub-pipelined design for KLEIN block cipher. There is a total of twenty four 64-bits registers introduced, which are using to encrypt and pass the data. These registers are also used for the temporary storage of data during the execution of each round. The pipelined registers carry data from one stage to the next. Only one operation is performed actually one place of the pipeline at a moment. Therefore, any operation is completed between registers of pipelines. It decides the execution time for each operation.

The design strategy of full-sub pipelined architecture is to insert one extra register before each round of cipher in sub-pipelined architecture. Therefore, a total of thirty six registers are utilized from the first round to the last round. After a thirty six clock cycles, the output has started to come out continuously. Fig. 4 shows the full-sub pipelined design for KLEIN block cipher.

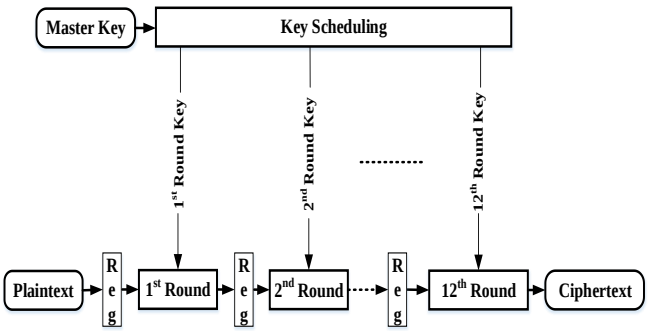

Fig. 2. Proposed Full-pipeline Structure

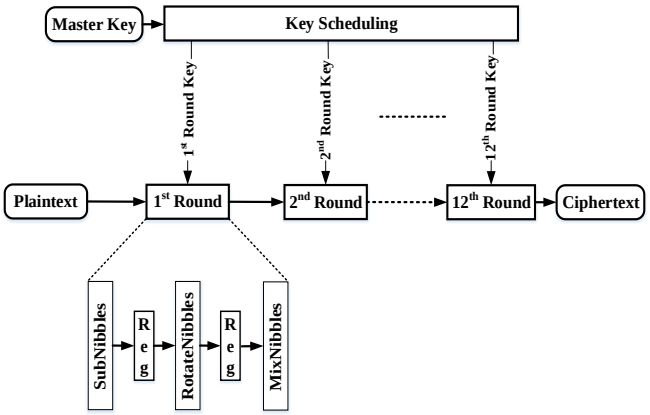

Fig. 3. Proposed Sub-pipeline Structure

In order to use a large number of pipelined registers, area computed in the terms of flip-flops, LUTs, and slices is increased to a large extent in the design. However, the performance of this architecture is improved in terms of throughput, since pipelining strategies enlarges the feasible frequency of applied clock cycle. Therefore, it adjusts the critical path delay. However, this design gives a good area-performance product. High throughput efficient design is achieved by selecting two main factors in the design. Initially, it is likely to fix an ideal quantity of pipelined points. Lastly, it is to choose the prime location for pipelining registers.

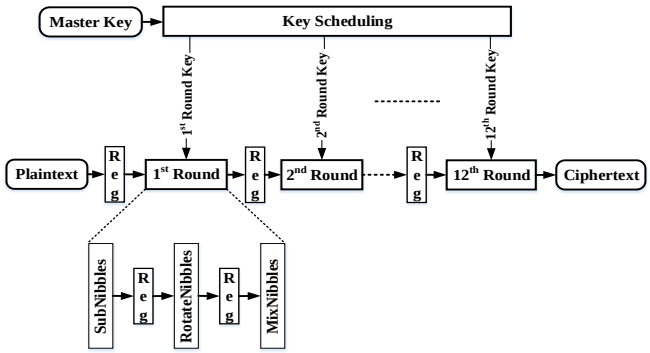

Fig. 4. Proposed Full-sub-pipeline Structure

B. *Multi-round based Architecture*

This design approach is fully based on round-based implementation, where multiple rounds can be completed within single clock cycle. For 64-bits key scheduling, ciphertext is generating after twelve rounds. By taking more than one round, ciphertext can be generated less than twelve cycles. For example, ciphertext can be generated within two clock cycles by completing six rounds into one clock cycle.

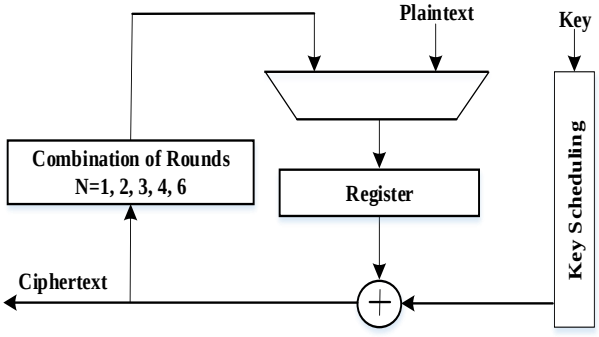

Fig. 5. Multi-round based Architecture for KLEIN Cipher

This proposed design can also be referred to as a multiple loop-rolling structure but it has made by combining more than one round into one loop. By doing this, ciphertext can be produced by decreasing the number of clock cycles. Therefore, it reduces the latency and increases the performance in terms of speed. In addition, the loop is executing through the round function and it also uses the same building blocks within one round. Control logic is set to high before the end of the last round so that data cannot be processed further starting from the loop.

IV. METHODOLOGY

A. *Measurement Domain*

The proposed designs are verified on different families of FPGAs platform like Spartan-6, Virtex-5, and Virtex-4. These families are utilizing various devices such as xc6slx16-3csg324, xc5vlx50t-3ff1136, and xc4vlx25-12ff668, respectively for making comparison. However, these designs are simulated by using ISim simulator and synthesized in Xilinx using ISE design suite 14.6, whereas power consumption are calculated with the help of Xilinx X Power Analyzer. The evaluations of hardware metrics are done after Post-Route simulation. In addition, each FPGAs family contains Configurable Logic Blocks (CLBs), which has a certain number of Flip-flops, LUTs, and Slices and configured in Configurable Logic Blocks (CLBs).

B. *Arrangements*

The proposed designs have used two strategies for KLEIN cipher evaluated on various hardware metrics. These architectures are simulated by using Xilinx synthesis tools. For result comparison, the proposed designs are configured as:

- Design_1: For this design, pipelined registers are introduced at starting of each round as configured in Fig. 2.
- Design_2: This designed is arranged by placing pipelined register in between of stages of each round as shown in Fig. 3.
- Design_3: Design_3 is configured by inserting pipelined registers by taking the concept of both designs Design_1 and Design_2 as mentioned in Fig. 4.
- Design_4: This design has implemented by combining either one or more than one rounds into one clock cycle as depicted in Fig. 5.

For fair comparison, these hardware implementations should be compared and verified under similar conditions like programming style, implementation process, synthesis tools, and FPGAs devices. Moreover, CLBs are examined alike for all devices, which may guide to best hardware implementation results concerning resource utilizations. Additionally, resources are concerned in terms of LUTs, Flip-flop, and Slices specified in all FPGA devices with different configurations.

C. *Measurement Metrics*

The proposed structures are measured on various hardware metrics like power consumption, speed and area-resources implemented in FPGAs platform. Performance metrics are classified based on the implementations. Some metrics are useful for testing the designs realized in hardware platforms, whereas others are helpful to examine the designs in software implementation. Lightweight ciphers are generally realized in hardware platforms, and a little part of them is also implemented on the software environment. In this way, hardware execution is the essential attention for lightweight ciphers.

Retrieval Number: B10221292S19/2019©BEIESP
DOI: 10.35940/ijitee.B1022.1292S19

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Table- I: FPGAs Implementation Results for different Ciphers

Designs	Latency (Cycles)	Slice	LUTs	Flip-flops	FMax (MHz)	Throughtput (Mbps)	Total Power (mW)	ENE/bit (nJ/bit)
Virtex-4 (xc4vlx25-12ff668)								
PRESENT80[22]	133	124	215	153	375.66	180.77	245.78	5.439
PRESENT128[22]	136	152	265	201	364.56	171.56	248.02	5.783
Virtex-5 (xc5vlx50t-3ff1136)								
PRESENT80[22]	133	67	190	153	542.30	260.96	562.75	8.626
PRESENT128[22]	136	73	239	201	431.78	203.19	562.67	11.077
Spartan-6 (xc6slx16-3csg324)								
PRESENT80[22]	133	48	170	153	257.40	123.86	21.61	0.698
PRESENT128[22]	136	61	220	201	210.66	99.13	21.76	0.878
Spartan-3 (xc3s50-5)								
LED80[24]	32	198	379	74	87.63	175.30	-	-
LED128[24]	48	227	444	76	87.63	116.54	-	-
PRESENT128[26]	256	117	159	114	114.80	28.46	-	-
HIGHT[26]	160	91	132	25	163.70	65.48	-	-
AES128[27]	534	393	531	338	-	16.86	23.83	-
AES128[28]	160	184	-	-	45.64	36.50	-	-
xTEA[29]	112	254	-	-	62.60	35.78	-	-
SIMON[30]	-	36	72	-	136	3.60	-	-

Hardware efficiency can be estimated from multiple points of view. Among them, area and speed requirements are the most significant parameters since small area requirement can limit both the cost and the power utilization efficiently. The speed of cipher is measured with regard to the throughput metric, which is evaluated by multiplication the block size with operational frequency followed by dividing the latency. It has turned out to be normal to utilize the term hardware efficient as an equivalent word for small area demands. However, area-resources is compared by consuming number of slices, LUTs, and flip-flops in the design. Moreover, performance in terms of speed is measured by calculating the latency in number of clock cycles and throughput in Mega-bits-per-second (Mbps). Mostly, registers used by lightweight ciphers for storing the data state and the key are the main reason for area consumption during hardware implementation. Moreover registers commonly comprise of flip-flops which have a somewhat high area and power challenges.

Equation (2) and equation (3) are used for calculating throughput and energy per bit and are measured in Megabit per second (Mbps) and Nano Joule per bits (nJ/bits), respectively [21].

$$\text{Throughtput} = \text{Block\_size X Operating Frequency} \Big/ \text{Latency} \quad (2)$$

$$\text{Energy / bit} = \text{Total Power X Latency} \Big/ \text{Operating Fregency X Block\_size} \quad (3)$$

V. RESULTS AND DISCUSSION

Table I and II communicate the result comparison for KLEIN and state-of-art lightweight block ciphers implemented on FPGAs platform. Mostly, these results are compared based on hardware performance like operating frequency, throughput, latency, Slices, Flip-flops, LUTs, and power consumption. Area metric is measured with the help of Slices, LUTs, and Flip-flops, whereas speed is concerned in terms of latency, throughput, and operating frequency. Moreover, power and energy are expressed by total power and energy per bit. As number of pipelined registers and width are increased, number of Flip-flops in area metric rises. In addition, if various CLBs are used for storing and processing the data, number of LUTs and slices can also be increased.

Among proposed design implementations, full-pipelined architecture demonstrates high throughput for all devices by modifying the critical path delay. This design utilizes less area in terms of Flip-flops and Slices compared to other pipelined structures. All encryption rounds and pipelined registers are inserted in hardware implementation. In addition, performance comparison examines that power consumption is almost constant for the same FPGA devices across all proposed architectures. Moreover, full-pipelined approach achieves throughput of 1474.56, 2414.13, and 2374.03 Mbps for xc6slx16-3csg324, xc5vlx50t-3ff1136, and xc4vlx25-12ff668 devices, respectively. Table I and Table II makes comparison of operating frequency and throughput results for proposed KLEIN cipher and state-of-art ciphers on different FPGAs devices. It indicates that multi-round based architecture has better energy consumption with highest throughput for Virtex-5 device in FPGAs.

Table- II: FPGAs Implementation Results for KLEIN Cipher

Designs	Latency (Cycles)	Slice	LUTs	Flip-flops	FMax (MHz)	Throughtput (Mbps)	Total Power (mW)	ENE/bit (nJ/bit)
Virtex-4 (xc4vlx25-12ff668)								
Design_1	12	1508	2921	768	445.13	2374.03	262	0.021
Design_2	24	1956	3054	3072	651.34	1736.91	283	0.163
Design_3	36	2298	3054	3840	666.02	1184.03	276	0.233
Design_4 (N=2)	7	428	751	131	214.39	1960.14	245	0.125
Design_4 (N=3)	4	572	1034	132	158.54	2536.64	242	0.095
Design_4 (N=4)	3	709	1292	130	123.66	2638.08	250	0.095
Design_4 (N=6)	2	986	1834	131	85.12	2723.84	245	0.090
Virtex-5 (xc5vlx50t-3ff1136)								
Design_1	12	890	2482	768	452.65	2414.13	568	0.044
Design_2	24	961	2812	3056	644.33	1718.31	576	0.335
Design_3	36	1120	2796	3840	712.30	1266.31	577	0.456
Design_4 (N=2)	7	206	554	131	246.77	2256.18	563	0.250
Design_4 (N=3)	4	277	745	132	175.62	2809.92	562	0.200
Design_4 (N=4)	3	362	953	130	136.74	2917.12	564	0.194
Design_4 (N=6)	2	471	1357	131	94.60	3027.2	564	0.186
Spartan-6 (xc6slx16-3csg324)								
Design_1	12	625	1874	768	276.48	1474.56	20	0.003
Design_2	24	871	2240	3072	412.86	1100.96	20	0.018
Design_3	36	913	2299	3840	396.17	704.30	20	0.028
Design_4 (N=2)	7	147	484	131	148.49	1357.62	20	0.015
Design_4 (N=3)	4	217	681	132	110.65	1770.40	20	0.011
Design_4 (N=4)	3	253	851	130	85.12	1815.89	20	0.011
Design_4 (N=6)	2	464	1257	133	59.25	1896.00	20	0.010

VI. CONCLUSION

Due to harsh efficiency-performance constraints, energy efficient, and high throughput implementations are required. Therefore, two different architectures are presented for the lightweight block cipher KLEIN. The full-pipelined configuration accomplishes a high throughput of 2414.13 Mbps for Virtex-5 family but also consumes high area compared to the state-of-art structures. Moreover, this design may be used in high bandwidth applications and high-end smart devices. However, six-stage multi-round based implementation achieves an energy efficient design than the pipelined architecture with high throughput. This proposed design consumes 0.186 nJ/bit with a throughput of 3027.20 Mbps for Virtex-5 family. The rolled strategy of multi-round architecture minimizes the total combinatorial cost, which is energy efficient for sensor nodes is Wireless Sensor Network (WSN).

REFERENCES

1. R. M. Davis, “The Data Encryption Standard in Perspective,” IEEE Commun. Soc. Mag., vol. 16, no. 6, pp. 5–9, 1978.
2. J. Daemen, V. Rijmen, and K. U. Leuven, “AES Proposal : Rijndael,” Complexity, pp. 1–45, 1999.
3. K. A. McKay, L. Bassham, M. S. Turan, and N. Mouha, “Report on lightweight cryptography,” Natl. Inst. Stand. Technol., vol. NISTIR 811, p. 26, 2017.
4. N. Hanley and M. O’Neill, “Hardware comparison of the ISO/IEC 29192-2 block ciphers,” Proc. - 2012 IEEE Comput. Soc. Annu. Symp. VLSI, ISVLSI 2012, pp. 57–62, 2012.
5. A. Bogdanov et al., “PRESENT : An Ultra-Lightweight Block Cipher,” Springer Berlin Heidelb., pp. 450–466, 2007.
6. D. Hong et al., “HIGHT: A New Block Cipher Suitable for Low-Resource Device,” Cryptogr. Hardw. Embed. Syst. CHES 2006, Springer, LNCS, vol. 4249, pp. 46–59, 2006.
7. K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, “Piccolo: An Ultra-Lightweight Blockcipher,” pp. 1–16.
8. L. Li, B. Liu, and H. Wang, “QTL: A new ultra-lightweight block cipher,” Microprocess. Microsyst., vol. 45, pp. 45–55, 2016.
9. W. Zhang, Z. Bao, D. Lin, V. Rijmen, B. Yang, and I. Verbauwhede, “RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms RECTANGLE,” Sci. China Inf. Sci., vol. 58, no. 12, pp. 1–15, 2015.
10. Z. Gong, S. Nikova, and Y. W. Law, “KLEIN: A new family of lightweight block ciphers,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 7055 LNCS, pp. 1–18, 2012.
11. T. P. Berger, J. Francq, M. Minier, and G. Thomas, “Extended Generalized Feistel Networks Using Matrix Representation to Propose a New Lightweight Block Cipher: Lilliput,” IEEE Trans. Comput., vol. 65, no. 7, pp. 2074–2089, 2016.
12. S. M. Sait and W. Hasan, “Hardware design and VLSI implementation of a byte-wise crc generator chip,” IEEE Trans. Consum. Electron., vol. 41, no. 1, pp. 195–200, 1995.
13. K. Rahimunnisa, P. Karthigaikumar, N. A. Christy, S. S. Kumar, and J. Jayakumar, “PSP: Parallel sub-pipelined architecture for high throughput AES on FPGA and ASIC,” Open Comput. Sci., vol. 3, no. 4, pp. 173–186, 2013.
14. G. P. Saggese, A. Mazzeo, N. Mazzocca, and A. G. M. Strollo, “An FPGA-based performance analysis of the unrolling, tiling, and pipelining of the AES algorithm,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 2778, pp. 292–302, 2003.

Retrieval Number: B10221292S19/2019©BEIESP
DOI: 10.35940/ijitee.B1022.1292S19

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

15. A. Kaur, P. Bhardwaj, and N. Kumar, "FPGA Implementation of Efficient Hardware for the Advanced Encryption Standard," Int. J. Adv. Res. Comput. Eng. Technol., no. 3, pp. 186–189, 2013.
16. A. Hodjat and I. Verbauwhede, "A 21. 54 Gbits / s Fully Pipelined AES Processor on FPGA," pp. 12–13.
17. K. U. Järvinen, M. T. Tommiska, and J. O. Skyttä, "A fully pipelined memoryless 17.8 Gbps AES-128 encryptor," ACM/SIGDA Int. Symp. F. Program. Gate Arrays - FPGA, pp. 207–215, 2003.
18. S. M. Yoo, D. Kotturi, D. W. Pan, and J. Blizzard, "An AES crypto chip using a high-speed parallel pipelined architecture," Microprocess. Microsyst., vol. 29, no. 7, pp. 317–326, 2005.
19. C. Wang and H. M. Heys, "Using a pipelined S-box in compact AES hardware implementations," Proc. 8th IEEE Int. NEWCAS Conf. NEWCAS2010, pp. 101–104, 2010.
20. H. Anwar, M. Daneshtalab, M. Ebrahimi, J. Plosila, and H. Tenhunen, "FPGA implementation of AES-based crypto processor," Proc. IEEE Int. Conf. Electron. Circuits, Syst., pp. 369–372, 2013.
21. D. H. Bui, D. Puschini, S. Bacles-Min, E. Beigne, and X. T. Tran, "AES Datapath Optimization Strategies for Low-Power Low-Energy Multisecurity-Level Internet-of-Things Applications," IEEE Trans. Very Large Scale Integr. Syst., vol. 25, no. 12, pp. 3281–3290, 2017.
22. C. A. Lara-Nino, A. Diaz-Perez, and M. Morales-Sandoval, "Lightweight Hardware Architectures for the Present Cipher in FPGA," IEEE Trans. Circuits Syst. I Regul. Pap., vol. 64, no. 9, pp. 2544–2555, 2017.
23. C. A. Lara-Nino, M. Morales-Sandoval, and A. Diaz-Perez, "Novel FPGA-Based Low-Cost Hardware Architecture for the PRESENT Block Cipher," Proc. - 19th Euromicro Conf. Digit. Syst. Des. DSD 2016, pp. 646–650, 2016.
24. W. Meier and D. Mukhopadhyay, "A Very Compact FPGA Implementation of LED and PHOTON," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 8885, pp. 304–321, 2014.
25. P. Singh, B. Acharya, and R. K. Chaurasiya, "High Throughput Architecture for KLEIN Block Cipher in FPGA," *2019 9th Annu. Inf. Technol. Electromechanical Eng. Microelectron. Conf.*, pp. 64–69, 2019.
26. P. Yalla and J. P. Kaps, "Lightweight cryptography for FPGAs," ReConFig'09 - 2009 Int. Conf. ReConFigurable Comput. FPGAs, pp. 225–230, 2009.
27. J. P. Kaps and B. Sunar, "Energy comparison of AES and SHA-1 for ubiquitous computing," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 4097 LNCS, pp. 372–381, 2006.
28. J. Chu and M. Benaissa, "Low area memory-free FPGA implementation of the AES algorithm," Proc. - 22nd Int. Conf. F. Program. Log. Appl. FPL 2012, pp. 623–626, 2012.
29. J.-P. Kaps, "Chai-Tea, Cryptographic Hardware Implementations of xTEA.".
30. A. Aysu, E. Gulcan, and P. Schaumont, "SIMON says: Break area records of block ciphers on FPGAs," IEEE Embed. Syst. Lett., vol. 6, no. 2, pp. 37–40, 2014.

AUTHORS PROFILE

Pulkit Singh received the Master's degree from the department of Electronics & Communication Engineering, National Institute of Technology, Allahabad, India in 2012. He is now a Ph.D. candidate in the department of Electronics & Communication Engineering, National Institute of Technology, Raipur, India. His main research interest lies in VLSI digital design, cryptography and network security, and embedded systems.

Piyush Modi received the Bachelor of Engineering degree in Electronics and Communication Engineering from Mahakal Institute of Technology, Ujjain in 2017 which is affiliated from Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal. He is presently pursuing Master of Technology degree in VLSI Design and Embedded system from National Institute of Technology, Raipur. His current research interest includes Network Security under Lightweight Cryptography, VLSI System Designs for Information Security and Hardware Security.

Bibhudendra Acharya received the M.Tech. and Ph.D. degree in Electronics and Communication Engineering from National Institute of Technology, Rourkela, India. He is currently serving as Assistant Professor in the Electronics and Communication Engineering Department, National Institute of Technology, Raipur, India. His research interest includes Cryptography and Network Security, Microcontroller and Embedded system, Signal Processing, Mobile Communication. He has more than 75 research publications in National/internationals Journals and conferences.

Rahul Kumar Chaurasiya received his Master's degree from the department of System Science and Automation, Indian Institute of Science, Bangalore, India and Ph.D. degree from the department of Electrical Engineering, National Institute of Technology, Raipur, India. He was a Senior Software Engineer with Brocade Communications Systems, Bangalore, in 2011-12. During 2013-18, he was Assistant Professor at the National Institute of Technology, Raipur. Since 2019, he is with the Malviya National Institute of Technology Jaipur as Assistant Professor. His areas of interest are pattern recognition, machine learning, data mining, brain-computer interface, optimization, and data structures. He has published a great deal of research papers at national and international journals, as well as conference proceedings. He is life member of IETE and IAN (Indian Academy of Neuroscience).