Parameterized AES-based Crypto Processor for FPGAs

Hassan Anwar[*♀], Masoud Daneshtalab[♀], Masoumeh Ebrahimi[♀], Sergei dytckov[♀], Juha Plosila[♀],
Hannu Tenhunen[♀], Giovanni Beltrame[*]

[♀] Department of Information Technology, University of Turku, Finland
[*] Ecole Polytechnique Montreal, Montreal, Canada

hassanbinanwar@gmail.com, Giovanni.Beltrame@polymtl.ca
{masdan, masebr, juplos, hannu.tenhunen, sergei.dytckov}@utu.fi

Abstract — **In this paper, we propose a parameterized crypto co-processor based on Advanced Encryption Standard (AES). This parameterized AES module is combined with a 32-bit general purpose 5-stage pipelined MIPS processor. The AES module used in this paper is fully pipelined. The processor fetches an instruction from the instruction memory and sends it to the decode stage. If the instruction is the crypto instruction it is pushed into the AES module during the decode stage. However if the instruction belongs to the MIPS processor, the remaining cycles will be completed on the MIPS processor. The parameterized AES module has different latencies on different rounds of AES according to the application requirements. The effects of different number of rounds on latency, memory, and area are studied and reported.**

Keywords—Cryptographic; Field Programmable Gate Array; Parameterized AES pipeline; Processor

I. INTRODUCTION

Cryptography is one of the major applications using Field Programmable Gate Arrays (FPGAs) as a platform to test the performance of an algorithm. For instance, web-servers and online banking system that needs to handle lots of encrypted authentications are benefited using FPGAs. Advanced Encryption Standard (AES) is a very well-known symmetric-cryptography algorithm for encrypting data. FPGAs successfully meet a required performance for this algorithm. Achieving a better Performance and speed are not the only reasons to select FPGA. There are some other characteristics of FPGAs like device utilization and availability in the market that makes them suitable choices for cryptography [1]. The main purpose of a cryptographic algorithm is to provide data security. Data Encryption Standard (DES) [2] had been used as a cryptographic algorithm for last numerous years but it is replaced by the Rijndael algorithm because of its failure to handle big input data. The Rijndael algorithm has become as a standard in cryptography and named Advanced Encryption Standard [3]. Encryption is a transformation procedure used to change the plain text to an unreadable form called ciphertext. Cryptographic algorithms implemented on hardware are more secure than the software implementation [4]. Field Programmable Gate Arrays (FPGAs) provide the finest platform for the hardware implementation of cryptographic algorithms because of their hardware adaptability nature and their capability to alter the design at any time. FPGAs provide the best solution for scientists to implement and explore the designs. When the Rijndael algorithm has established as a standard encryption algorithm, many hardware implementations have been developed on FPGAs and ASICs [4-11]. Software implementations are also available due to their flexibility, convenience and inherited ease of up gradation [12]. ASIC provides a low power design but lacks of flexibility and short time to market [22-25]. This paper proposes an approach to combine the general purpose processor with the crypto co-processor. In this work, general purpose processor is MIPS-32 [13] with five pipelined stages while the crypto processor utilizes a parameterized AES-128, 192, and 256 as a crypto algorithm.

Our design is implemented in such a way that crypto instructions do not halt the instruction fetch cycle of the processor even though the crypto co-processor is running on that time. Each instruction is fetched from the instruction memory unit and it completes all cycles on the MIPS processor if the instruction is designed for the processor. If the instruction is not a MIPS instruction, it will be sent to the AES crypto co-processor after the following clock cycle of the decode phase. We incorporate a parameterized AES module as a crypto co-processor with MIPS. The main aim to use a parameterized AES is to select the number of rounds according to the application requirements. Moreover, AES is the crypto co-processor which do not affect the MIPS processor pipeline stages.

This paper is organized as follows. Section II presents the AES algorithm. Section III describes the FPGA implementation of AES while the pipelined version of AES is presented in Section IV. The MIPS processor and the integration of the parameterized AES in MIPS have been described in Sections V and VI, respectively. In Section VII experimental results are discussed while Section VIII concludes the paper.

II. AES ALGORITHM

Advanced Encryption Standard (AES) based on the Rijndael algorithm [1] has become a standard in 2001. AES supports different data sizes of 128, 192 and 256 bits. It also has the ability to support different key sizes of 128, 192 and 256 bits. Encryption and decryption depends on the number of rounds, i.e. 10-rounds (needs 128 bits), 12-rounds (needs 192 bits), and 14-rounds (needs 256 bits). The AES algorithm involves four different steps as Add Round Key , S-box substitution, Shift Rows, and MixColumn (see Fig. 1) [35]. S-box substitution is the only non-linear transformation which involves two steps. The first step is the multiplicative inverse of input data bytes then an affine transformation is applied on it. Two different ways to implement S-box entries either by using look up tables or computing mathematically. Both linear operations (i.e. Shift Rows and MixColumn) are used to create diffusion.

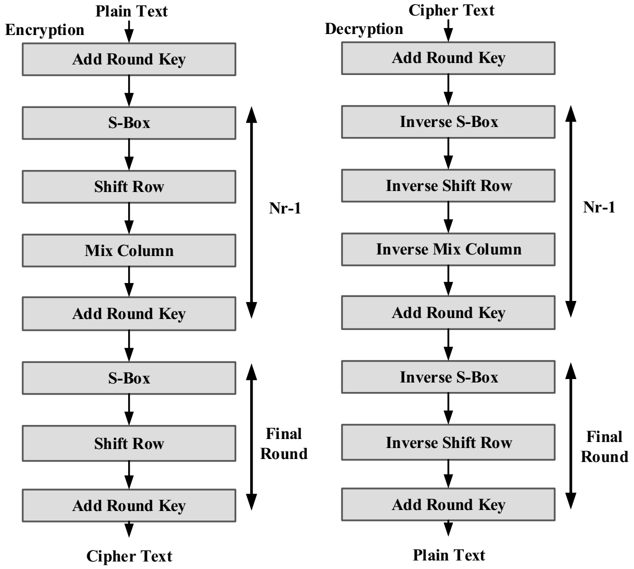

Fig. 1. AES Algorithm.

A. *Add Round Key*

Key addition layer is used to produce the number of keys used in each round of encryption and decryption. The key addition layer has two inputs. First is the output of the MixColumn and the second is the sub-key. This layer is simply formed by a bit wise XOR gate between binary inputs

B. *S-Box design (Subbyte and Inv Subbyte)*

S-Box is the non-linear operation of the AES algorithm, which is used to generate confusion in the input bytes. There are two different methods to implement S-Box. The first method is to produce a look up table of 256-by-8 bit with stable entries; but critical path is too long in this case with less hardware resources [14]. The second method is to use an inversion Galois Field approaches (GF (2^8)). This method achieves short critical path but costs hardware consumption [14]. The decryption of data contains similar multiplicative inversion and inverse affine transformation steps.

C. *Shift Rows and Inv Shift Rows*

Rows shifting and column mixing are used to add diffusion. The shift rows operation is used by encryption while the inverse shift rows operation is used by decryption.

D. *MixColumn and Inv MixColumn*

MixColumn is the next step of the diffusion layer. It is the linear transformation step which combines each column of the matrix. In this step, each column of the state matrix is multiplied by a specific matrix. Both multiplication and addition of the coefficient are done in GF (2^8). Decryption uses Inverse MixColumn operation, in order to inverse the output of MixColumn step.

III. FPGA IMPLEMENTATION OF AES

The effective implementation of the AES algorithm on FPGA is being under discussion from last numerous years in terms of throughput, minimum area usage, and high speed. The central motive to select FPGA for the implementation of the cryptographic algorithms is to permit altering designs with almost no extra time. A low cost and a short design cycle are also the FPGA design attributes. An FPGA-based AES implementation is presented in [4] while several other high speed implementations have been explored in [5], [6], [7], [8], [9] and [10] in which the speed ranged from Mbps to several Gbps. Some of the presented AES approaches have implemented AES as a pipeline manner either in inner rounds or outer rounds. The processor-based implementation is presented in [11] and [12]. The architecture with a smaller data path is presented in [14] which can achieve 20% reduction in the area overhead. 15% area reduction is achieved by [15] while only 0.052 mm^2 area required by [16] to support both encryption and decryption and to obtain the throughput of 311 Mbps. In [17] a method is presented with the throughput of 2.2 Mbps while consuming only two memory blocks and 124 slices. Optimizing S-Box is another way to optimize the AES implementation in terms of area and speed [12], [18], [19], [20], [21] and [22]. The optimization of MixColumn and inverse MixColumn is also investigated in [23], [24] and [25] while the pipeline implementation of AES on FPGA is proposed in [26].

IV. PIPELINED AES

Fig. 2 shows the implementation of the pipelined AES in which several procedures can be run in parallel [36]. Pipelining in the Advanced Encryption Standard is used to get a higher throughput and a higher speed as multiple rounds of AES can be handled concurrently. However, area consumption is very high [27]. Pipelining, sub-pipelining, and loop unrolling are the techniques used to increase the speed of the AES algorithm. The optimization techniques are not efficient solutions to increase the speed [28]. The cost of hardware resources and their processing speeds are very slow in unrolled AES architectures [29]. Pipelining and sub-pipelining provide high throughput [30] up to 26.64 Gbps. Since pipelining methods consume more hardware resources, researchers have been focusing on the implementation of the AES pipelining in a way that it consumes less hardware resources. Several pipelined methods have been proposed to achieve high throughput and low area overhead [31] and [32]. The throughput achieved by [32] is around 20.3 Gbps. In this paper, we present a fully pipelined implementation of 128, 192, and 256 bits AES. Our design is implemented on Virtex 6 ML605 and the design is run at the frequency of 1042 MHz.

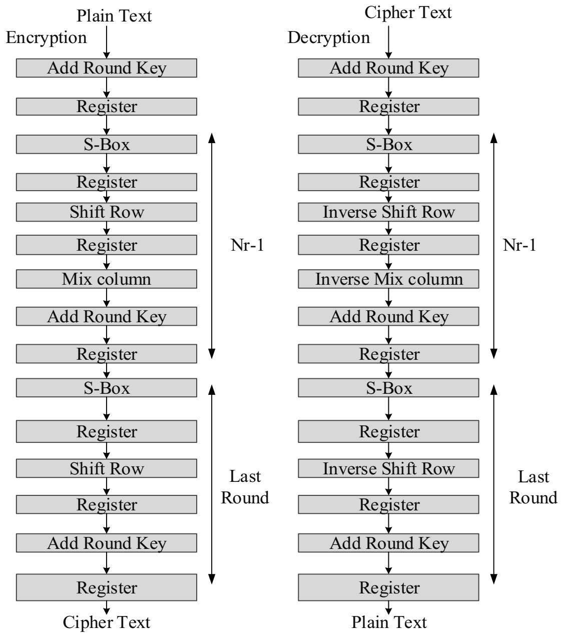

Fig. 2. Pipelined AES.

V. MIPS PROCESSOR

Fig. 3 shows the MIPS in a five-stage pipelined processor [36]. Pipelining increases the number of concurrent running instructions which in turn increases the frequency and performance. This means that the data path is separated into unalike pieces named:

- Instruction Fetch
- Instruction Decode
- Execution or address calculation
- Data Memory access
- Write Back

MIPS is a 32-bit processor used to execute R-type, J-type, and I-type instructions [33]. The instruction format is shown in Fig. 4. MIPS instructions are word-aligned, meaning that the first byte of the instruction is stored in an address divisible by four. MIPS is a register-based with load/store architecture which is commonly referred to as the RISC architecture.

The Instruction fetch unit has an instruction memory module and a program counter module to read an instruction from the instruction memory and to send these contents to the decode stage which has read and write registers to read an instruction based on operands. The Execution unit is used to execute a particular instruction and places the result on the next pipeline stage. The data memory unit contains a memory unit to write the data on a particular address and the write back unit places the result back to the register file.

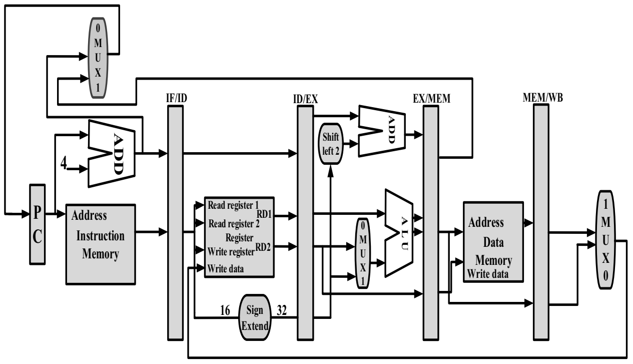

Fig. 3. Data Path of MIPS.

R- Type

Opcode[31-26]	rs[25-21]	rt[20-16]	rd[15-11]	shamt[10-6]	function[5-0]

I- Type

Opcode[31-26]	rs[25-21]	rt[20-16]	Address / immediate[15-0]

J- Type

Opcode[31-26]	Target address [25-0]

Fig. 4. Instruction format.

VI. MIPS WITH PARAMETERIZED AES

In the proposed design shown in Fig. 5, we integrate the parameterized crypto co-processor based on the AES algorithm with the MIPS processor in such a way that AES is executed as the crypto co-processor. The hardware implementation using FPGA provides a significant performance gains compared to software implementation using the general purpose processor (microprocessor) in terms of parallel processing, pipelining, word size and speed. More details are shown in Table I and Table II.

Table III provides the information about the development process in terms of language, cycle, tools, and maintenance. Pipelining and parallel processing are very restricted on the general purpose processor. Throughput up to several Gbps can be easily attained by using FPGA. The Crypto algorithm agility is also possible by using FPGA. Pipelining and parallel processing of AES are very limited on general processors because of their internal structures and fixed sizes of functional units [34][35]. A general purpose processor is not suitable for the implementation of the cryptographic algorithms because of two reasons.

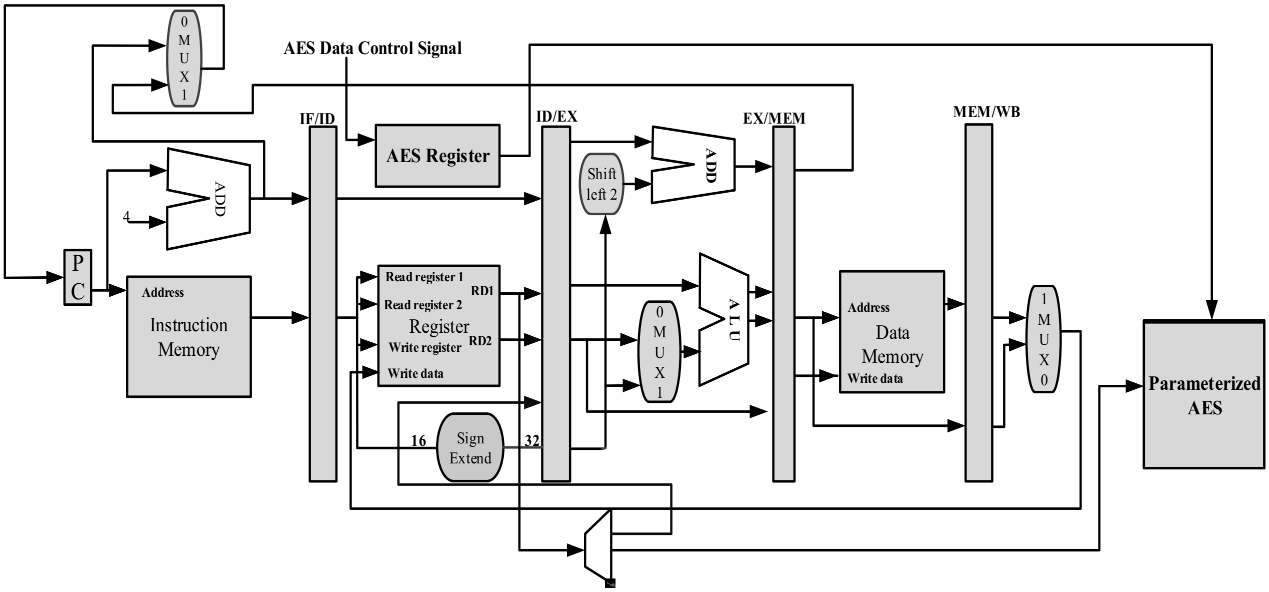

Fig. 5. Parameterized crypto processor with Processor.

First, their storage capability is limited to store key(s) in the internal register of the processor [34]. Second, if there is an attack on cache, it can affect the cryptographic algorithm. The basic entity used to implement encryption and decryption is shown in Fig. 6. The combinational logic circuit is used to implement one round of cipher with the support of single multiplexer and a register. Two main characteristics of this architecture are as follows: one block of data is encrypted in one clock cycle and the number of clock cycles necessary to encrypt a single block of data is equal to cipher rounds.

Table I. Performance Characteristics of Cryptographic algorithm on FPGA and Microprocessors

Performance Characteristics	FPGA	Microprocessor
Parallel Processing	Potential	Restricted
Pipelining	Potential	Restricted
Word Size	Adaptable	Static
Speed	Fast	Adequate Fast

We propose a design called parameterized AES in which the number of rounds can be parameterized at design time according to the application requirements and constraints. The structure of the proposed design is depicted in Fig. 7.

Table II. Functionality of Cryptographic algorithm on FPGA and Microprocessors.

Functionality	FPGA	Microprocessor
Algorithm Agility	Potential	Potential
Tamper Resistance	Restricted	weak
Access control to Keys	moderate	weak

Table III. Development process of Cryptographic algorithm on FPGA and Microprocessors.

Development Procedure	FPGA	Microprocessor
Description Language	VHDL	Restricted
Design Cycle	moderately long	Short
Design Tools	moderately expensive	inexpensive
Maintenance and upgrades	inexpensive	inexpensive

Table IV. Parameterized Rounds

AES	Number of Rounds	Parameterized Round (K)
AES-128	4	1,2,5,10
AES-192	6	1,2,3,4,6,12
AES-256	4	1,2,7,14

The basic difference between Fig. 6 and Fig. 7 is that the combinational part of the circuit implements K rounds, instead of a single round. K must be a divisor of the total numbers of rounds. For instance, if the design is based on AES-128, the parameterized rounds can be 1, 2, 5, and 10. The combinational part of the circuit constitutes the majority of circuit area while the total area of the parameterized AES increases with the number of rounds. Table IV shows the round values of different block sizes used in our design.

In the proposed design, the crypto instruction is fetched from the instruction memory of the MIPS processor and will be executed by the crypto co-processor (parameterized AES). Thereby, each crypto instruction has a constant prefix (e.g. 010011 in Table V) which is used by the decode stage of the processor to show that the execution of this instruction is on the crypto co-processor.

The crypto co-processor uses immediate and register instructions. The memory read address (mra) is the starting address of the AES memory. The AES memory unit shown in Fig. 5, where data is stored for the AES operations and the results are stored in the memory write addresses (mwa). The size of data is determined by length. The size of length in our design is 10 bit, so it ranges from 1 to 1024 blocks where each block can support 128 bits, 192 bits and 256 bits of data.

Table V : Crypto co-processor instruction format.

Immediate instruction	**Mra**	**mwa**	**length**	**ecn/dec**
6-bit				
010011	5-bit	5-bit	10-bit	6-bit
Register instruction	**Mra**	**mwa**	**length**	**ecn/dec**
6-bit				
010011	5-bit	5-bit	10-bit	6-bit

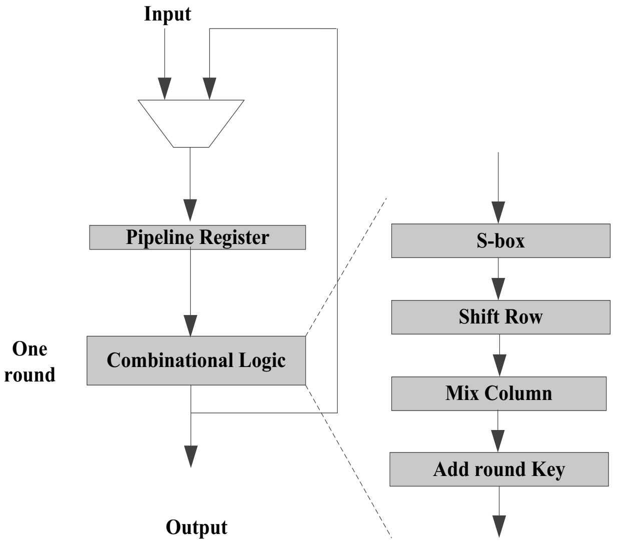

Fig. 6. Basic Iterative Architecture.

VII. EXPERIMENTAL RESULT

The proposed design is implemented on FPGA using Xilinx ISE 14.4. We evaluate the affects of different configurations (number of rounds) on area consumption, latency, memory and LUT. We run the system configured with different number of rounds for AES-128, AES-192, and AES-256. For each AES we have the same number of inputs (plaintext). After configuring the number of rounds and the AES type, we feed these inputs into the pipeline stages of the crypto co-processor to compute the latency and throughput for each AES with the configured number of rounds. Table VI summarizes all the details of AES-128, AES-192, and AES-256 with different number of rounds. Minimum area consumption is observed for AES-128 with 1-round AES module, and maximum area consumption is observed for AES-256 on 14-round AES module. The area (cost) totally depends on the number of rounds configured for the crypto co-processor.

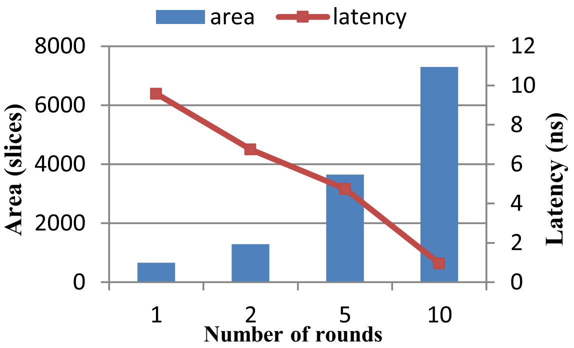

Fig. 8. Latency and area usage for different rounds of AES-128.

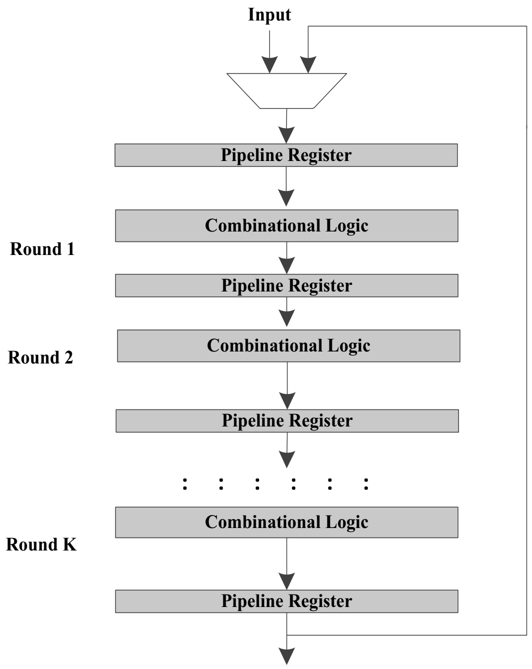

Fig. 7. Parameterized AES.

The effect of changing the number of rounds on memory and LUT usages for 1- and 2-round configurations of AES-128 is almost the same. The maximum memory and LUT usages are observed on 14-round configuration of AES-256.

Fig. 8, Fig. 9, and Fig. 10 illustrate the area overhead and latency of AES-128, AES-192, and AES-256, respectively, in terms of the number of configured rounds. As can be seen from the results, when the number of rounds increases, the cost (area) raises significantly while the latency is reduced and throughput is increased accordingly. As shown in Fig. 11, Fig. 12, and Fig. 13, throughput for AES-128, AES-192, and AES-256 are 30 Gbps, 42 Gbps, and 64 Gbps, respectively. Fig. 14 illustrates the area consumption of MIPS with AES-128.

Table VI. Comparison of AES-128, AES-192 and AES-256

Block size	128				192				256			
Number of Rounds	Latency (ns)	Area	LUT	Memory usage KB	Latency (ns)	Area	LUT	Memory usage KB	Latency (ns)	Area	LUT	Memory usage KB
1	9.59	662	369	230564	11.508	5494	2623	277244	13.5	5228	4185	267300
2	6.75	1290	746	232420	8.71	6122	3000	289444	11	7068	5166	278500
3	-	-	-	-	4.79	6865	3466	300644	-	-	-	-
4	-	-	-	-	3.58	7843	4444	311844	-	-	-	-
5	4.75	3645	4774	272740	-	-	-	-	-	-	-	-
6	-	-	-	-	2.78	9455	5502	320932	-	-	-	-
7	-	-	-	-	-	-	-	-	6.75	11023	7741	334500
8	-	-	-	-	-	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-	-	-	-	-	-
10	0.959	7290	9548	545480	-	-	-	-	-	-	-	-
11	-	-	-	-	-	-	-	-	-	-	-	-
12	-	-	-	-	0.959	18910	11040	641864	-	-	-	-
13	-	-	-	-	-	-	-	-	-	-	-	-
14	-	-	-	-	-	-	-	-	0.959	22046	15482	669000

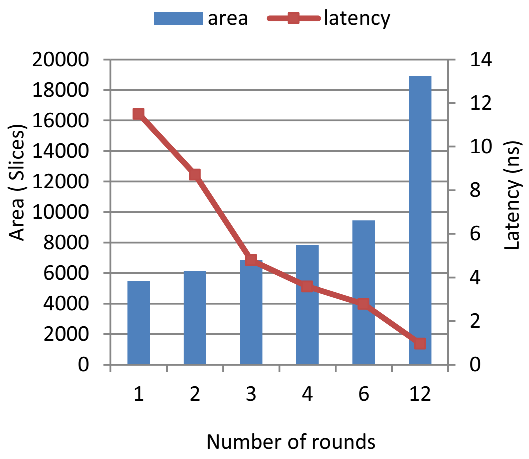

Fig. 9. Latency and area usage for different rounds for AES-192

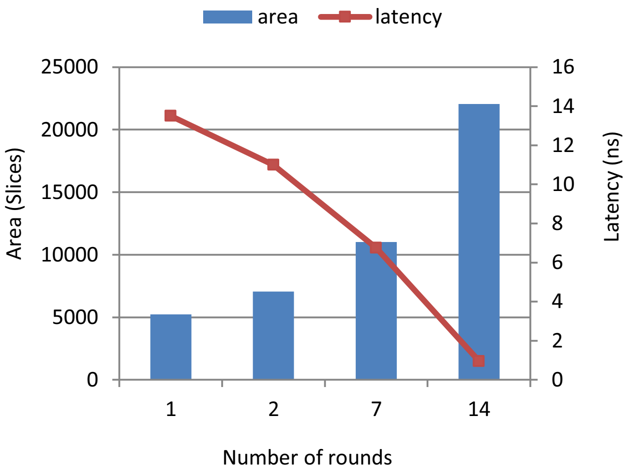

Fig. 10. Latency and area usage for different rounds for AES-256

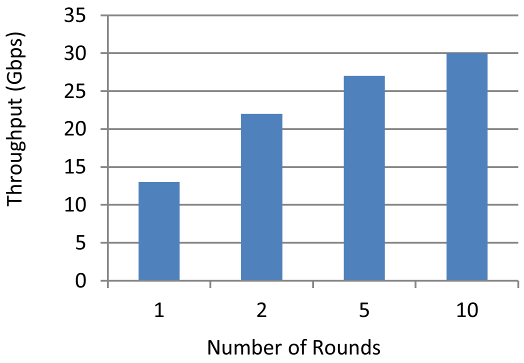

Fig. 11. Throughputs for AES-128

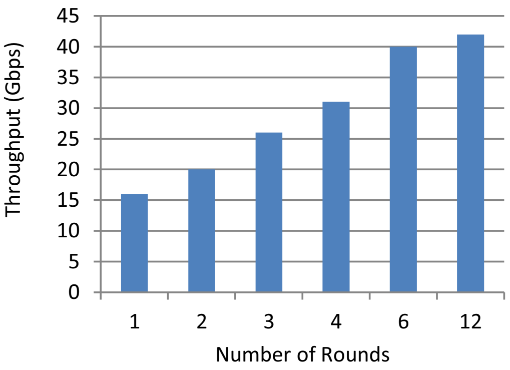

Fig. 12. Throughputs for AES-192

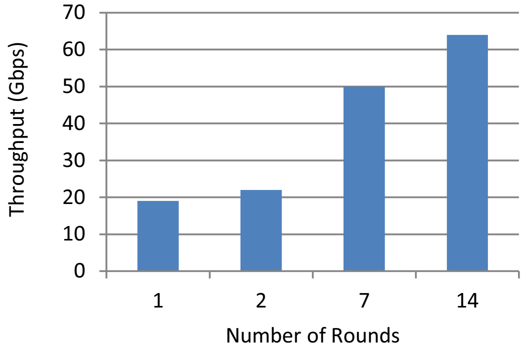

Fig. 13. Throughputs for AES-256

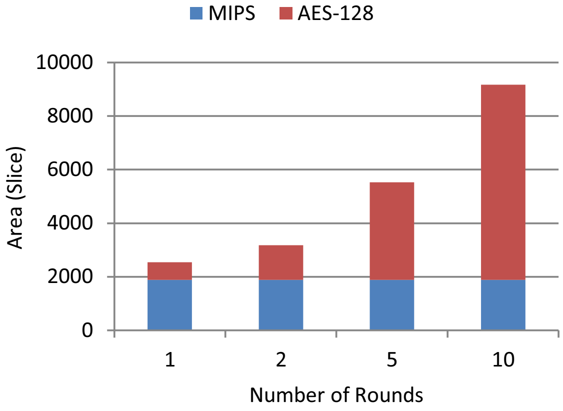

Fig. 14. MIPS with AES-128 Area Consumption

VIII. CONCLUSION

In this paper, we proposed a parameterized crypto co-processor. The parameterized AES can encrypt and decrypt data according to the application requirements. If the requirement is to have a low-latency design, the number of rounds should be increased while if the constraint is the area utilization, the number of rounds should be reduced. Experimental results are performed to explore the effect of the round parameter on area, latency, and throughput.

IX. ACKNOWLEDGEMENT

This work was supported by VINNOVA (Swedish Agency for Innovation Systems) within the CUBRIC and ERoT projects, and academy of Finland.

REFERENCES

[1] 40Gbit AES Encryption using OpenCL and FPGAs, http://nallatech.com/images/stories/technical_library/white-papers/40_gbit_aes_encryption_using_opencl_and _fpgas_final.pdf.

[2] B. Schneier, Applied Cryptography, Wiley, New York, 1996.

[3] National Institute of Standard and Technology (USA, Advance Encryption Standard). FIPS 197, Available at, http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf, September 1999.

[4] S. M. Yoo, D. Kotturi, D. W. Pan and J. Blizzar, "An AES crypto chip using a high-speed parallel pipelined architecture," Elsevier, Microprocessor and Microsystem, vol. 29, pp. 317-236, January 2005.

[5] P. Chodowiec, "Comparison of the hardware performance of the AES candidates using rconfigurable hardware," Master's thesis, George Mason University, March 2002.

[6] H. Kuo and I. Verbauwhede, "Architectural optimization for 1.82 Gbits/sec VSLI implementation of the AES Rijindael algorithm," Cryptographic Hardware and embedded systems (CHES'01), vol. 2162, pp. 51-64, Springer-Verlag, 2001.

[7] N. Sklavos and O. Koufopavlou, "Architecture and VLSI implementation of the AES-proposal Rijndael," IEEE Trans. Computers, vol. 51, pp. 1454-1459, 2002.

[8] P. R Schoumont, H. Kuo and I. M. Verbauwhede, "Unlocking the design secrets of a 2.29 Gb/s Rijndael processor," ACM Conference on Design Automation (DAC 2002), pp. 634-639, 2002.

[9] S. Morioko and A. Satoh, "A 10 Gbps full-AES crypto design with twisted BSS s-box architecture," IEEE International Conference on Computer Design VLSI in Computers and Processors, pp. 98-103, 2002.

[10] U. Mayer, C. Oelsner and T. Kohler, "Evaluation of different Rijndael implementaion for high end servers," IEEE International Symposium on Circuits and Systems ,vol. 2, pp. 348-351, 2002.

[11] S. Tillich and J. GroBschadl, "Instruction set extension for efficient AES Implementation on 32-bit processor," proceding. 8th International workshop in cryptographic Hardware and Embedded System, Yokohama, Japan, LNCS 4249, ISBN 978-3-540-46559-1, pp. 270-284, 2006.

[12] S.Tillich and J. GroBschadl, "Accelerating AES using Instruction set extensions for elliptic curve cryptography," proceding. International Conference on Computional Science and its Applications , LNCS3483, ISBN 978-3-540-25863-6, Singapore, pp. 665-675, 2005.

[13] MIPS 32 Architecture for programmers, vol . I, "Introduction to MIPS Architecture," Available at http://www.mips.com/products/product-materials/processor/mips-architecture/2008.

[14] V. Fischer, M. Drutarovsky, P. Chodowiec and F. Gramain, "Inv mixcolumn decomposition and multilevel resource sharing in AES implementations," IEEE trans. VLSI syst, vol. 13, pp. 989-992, 2005.

[15] A. C Zigiotto and R. d'Amore, "A low-cost FPGA implementation of the advance encryption standards algorithm," Symposium on Integrated Circuits and System Design, pp. 191-196, 2002.

[16] A. Satoch, S. Morioka, K. Takano and S. Munetoh, "A compact Rijndal hardware architecture with s-box optimization," Theory and Application of Cryptology and Information Security, vol. 2, pp. 239-254, Springer-Verlag, 2001.

[17] T. Good and M. Benaissa. "AES FPGA from fastest to smallest," International Workshop on Cryptographic Hardware and Embedded System, vol. 3659, pp. 427-440, Springer-Verlag, 2005

[18] S. McMillan and C. Patterson, "JBits implementation of the advance encryption standard(Rijndael)," Filed–Programmable

Logic and Applications, vol. 2147, pp. 162-171, Springer-Verlag, 2001.

[19] Amphion Documentation of Cryptographic cores. Available at http://www.amphion.com.

[20] Helion Documentation of Cryptographic cores. Available at http://www.heliontech.com.

[21] A. Rudra, P. K. Dubey, C. S. Jutla, V. Kumar, J. Rao and P. Rohatgi, "Efficient Rijndael encryption implementation with composite field arithmetic," International workshop on Cryptographic Hardware and Embedded System, vol. 2162, pp. 171-184, Springer-Verlag, 2001.

[22] J. Wolkerstorfer, E. Oswald and M. Lamberger, "An ASIC implementation of the AES s-boxes," The Cryptographer's Track of the RSA Security Conference, vol. 2271, pp. 67-78, Springer-Verlag, 2002.

[23] S. Morioko and A. Satoh, "An Optimized s-box circuit architecture for low power AES design," International workshop on Cryptographic Hardware and Embedded Systems, vol. 2523, pp. 172-186, Springer-Verlag , 2002.

[24] N. Mentens, L. Batina, B. Preneel and I. Verbauwhede, "A systematic evaluation of compact hardware implementation for the Rijndael s-box," CT-RSA'05, vol. 3376, pp. 323-333, Springer-Verlag, 2005.

[25] J. Wolkerstorfer, "An ASIC implementation of the AES mix column operation," Austroship 2011, pp. 129-132, Austria, 2011.

[26] V. Fischer and F. Gramain. "Resource sharing in Rijndael implementation based on a new mixcolumn and inv mixcolumn relation, " unpublished.

[27] N. C Iyer, P.V Anandmohan , D. V Poornaiah and V.D Kulkarni, "High through put, low cost, fully pipelined architecture for AES Crypto Chip," Annual IEEE India Conference, pp. 1-6, September 2006.

[28] Y. Zhang and X. Wang, "Pipelined implementation of AES encryption based on FPGA," IEEE International Conference on Information theory and Information Security, pp. 170-173, December 2010.

[29] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989. I. Verbauwhede, P. Schaumont and H. kuo, "Design and performance testing of a 2.29-Gb /s rijndael processor," IEEE J. Solid State Circuits, vol. 38, pp. 569-572, 2003.

[30] N. C Iyer, P. V Anandmohan , D. V Poornaiah and V. D Kulkarni, "High through put, low cost, fully pipelined architecture for AES crypto chip," Annual IEEE India Conference, pp. 1-6, September 2006.

[31] A. Dandalis, "A comparative study of performance of AES candidates using FPGA's," The third Advance Encryption Standard (AES3) Candidate Conference, April 2000.

[32] G.P Saggese, A. Mazzro, N. Mazzocca and A. Strollo. "An FPGA-based performance analysis of the unrolling , tiling, and pipelining of the AES algorithm," In proceeding. International conference on field-Programmable Logic and Applications, vol. 2778 of LNCS, pp. 292-302. Springer-Verlag, 2003.

[33] D. A Patterson and J. L Hennessy, " Computer Organization and Design, The Hardware / Software Interface," Second Edition, Morgan Kaufmann Publishers, Inc. 1998.

[34] L. Gaspar, V. Fisgher, L. Bossuet and M. Drutarovsky, "Cryptographic extension for soft general purpose processors with secure key management," IEEE International conference on Field Programmable Logic and Applications, pp. 500-505, 2011.

[35] M. Ebrahimi, A. Jahangireyan, "New analytical formulations for calculation of dispersion parameters of Gaussian model using parallel CFD", Journal of Environmental Fluid Mechanics, Vol. 13, 2, pp125-144. 2013.

[36] A, Hassan, et al. "Integration of AES on Heterogeneous Many-Core system." Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on. IEEE, 2014, pp. 424-427.

[37] Anwar, Hassan, et al. "FPGA implementation of AES-based crypto processor."Electronics, Circuits, and Systems (ICECS), 2013 IEEE 20th International Conference on. IEEE, 2013, pp. 369-372.