ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 7, July 2021

Design and Implementations of the Hummingbird Cryptographic Algorithm on FPGA

[1] Pooja R K, [2] Dr. Vijayalakshmi D
[1][2] Dept. of ECE, Bangalore Institute of Technology

Abstract: Hummingbird is a brand new ultra-light-weight cryptographic algorithm focused on useful resource-restricted gadgets like RFID tags, smart cards, and Wi-Fi sensor nodes. In this project, we describe effective hardware implementations of a stand-alone Hummingbird component in field -programmable gate array (FPGA) gadgets. We enforce an encryption core and an encryption/decryption core at the low-cost Xilinx FPGA collection Spartan-6 and evaluate our consequences with different mentioned light-weight block cipher implementations at the identical collection. Our experimental consequences spotlight that with inside the context of low-fee FPGA implementation. Hummingbird has complimentary performance and low area essentials.

Index Terms— Hummingbird, Light-weight cryptography, RFID.

INTRODUCTION

Hummingbird algorithm is one of the currently proposed light-weight cryptographic algorithms focused on useful resource-restricted gadgets like RFID (radio frequency identification), smart cards, and the majority of Wi-Fi sensor nodes. The primary benefit of this algorithm is that it affords good enough protection with smaller block size. As in step with the preceding works in this algorithm, area and performance are primary layout tradeoffs of this algorithm. Performance is multiplied via way of means of loop unrolling and area is optimized via way of means of looping. So optimization in each location and overall performance is a large challenge. This work proposes an efficient hardware structure for the hummingbird algorithm using partial loop unrolling which issues each area and overall performance of hardware implementation. The average structure changed into modeled the usage of Verilog HDL and synthesized the usage of cadence RTL compiler with 45nm Technology from TSMC. The proposed layout is additionally carried out in low fee Spartan three FPGA board and the consequences are in comparison with the present implementations. Results display that there may be a place discount of around 6% and throughput nearly get doubles. The significance of low-fee gadgets like RFID (Radio-frequency identification), smart cards, and diverse Wi-Fi sensor gadgets is growing in our modern-day living. So the safety of such gadgets could be very important. Since those gadgets are extraordinarily useful resource-restricted in phrases of computing strength, battery strength delivery, and memory, the standardized cryptographic algorithms consisting of AES(Advanced encryption standard), DES(Data encryption standard), that are properly targeted on software program implementation in preference to hardware, cannot be used as it's far in those gadgets. So a positive elegance of cryptographic algorithms called lightweight cryptography is evolved. The layout metrics of lightweight cryptography are security, cost, and performance.

RELATED WORKS

Many light-weight cryptographic algorithms are introduced till now, a lightweight algorithm named HIGHT is proposed with 3048 gate equivalents (GE) that are a great deal quicker than AES [1]. Scalable Encryption Algorithm (SEA), with a block length and key length of ninety-six bits, and phrase length of eight bits and ninety-three rounds operation can encrypt one information block inside 1428 clock cycles and 3758 GE[2]. Slight Modifications is completed on Classical Block Cipher light-weight DES variation referred to as DESL (DES Lightweight) [3]. ASIC implementation of the identical calls for 1848 GE and it can encrypt one sixty-four-bit information block in a hundred and forty-four clock cycles. The implementation of DESL has about 20% smaller chip length than DES. The key whitening approach may be beneficial to enhance the safety of cipher in the DESXL algorithm. For encrypting the plaintext, it calls for 2; one hundred seventy GEs and a hundred and forty-four clock cycles. The PRESENT consists of rules for a lightweight substitution, permutation primarily based totally block cipher, which

ISSN (Online) 2394-2320

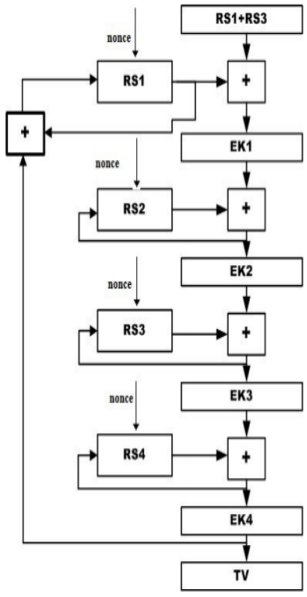
International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 7, July 2021

operates on 32 rounds, key length of eighty or 128 bits, and sixty-four-bit block length. The PRESENT serial model may be carried out with 1000GEs [4]. The Hummingbird algorithm is one of the current supplied ultra-lightweight cryptographic sets of rules [5]. The length of the important thing and the inner nation of Hummingbird afford a good enough protection stage for plenty of embedded applications. The existing researches are occurring for the improvement and special implementation of the identical due to the simplicity with inside the structure. Until now the hummingbird has been carried out on a target platform, software program in addition to hardware and indicates accurate performance in each. Xinxin fan carried out the algorithm on sixteen-bit in addition to 4-bit microcontroller.[6] Daniel Engels carried out the structure on the eight-bit microcontroller.[7] Xinxin fan carried out each location orientated and throughput orientated layout of a hummingbird on low fee FPGA.[6,8] Biao min proposed one more efficient hardware implementation of the identical FPGA.[9] Ismail san proposed but any other implementation on FPGA the usage of coprocessor approach.[10] All those implementations indicate that this algorithm works properly on different target platforms.

HUMMINGBIRD ALGORITHM

Hummingbird is a newly proposed ultra-light-weight cryptographic algorithm, combining a block cipher and a flow cipher. The layout of the hummingbird includes a 16-bit block length, 256-bit key length, and 80-bit inner state. This algorithm's primary benefit is that it has a smaller block length as compared to different algorithms and affords enough protection despite the fact that the block length is small. The common structure of the hummingbird cryptographic algorithm includes 4 16-bit block ciphers EKi (i=1,2,3,4), 4 16-bit inner state registers RSi (i= 1,2,3,4), and a 16-bit LFSR (linear feedback shift register). The 256-bit key with four inner registers and LFSR provides a good enough protection level. For every block EKi, 256-bit secret key's divided into 4 64-bit sub keys Ki (i=1, 2, 3, 4).



Notations Used	
RS1-RS4	Internal State registers to hold nonce value
EK1-EK4	State registers to hold the modulo addition values of 16 bit register RS with EK
+	Modulo addition
TV	Register to store the final output after 4 rounds of initialization

Fig. 1 Initialization

3.1 INITIALIZATION:

The initialization technique is shown in Fig.1 initializes the 4 internal states register EK1 to EK4 and get the LFSR primary value earlier than encryption starts. The 4 internal state registers are first loaded with 4 16-bit random NONCE values. Taking RS1 + RS3 as input data, 4-block ciphers are consecutively executed 4 times and the states are up to date consequently as shown in Fig.1. The final output after 4 iterations is inside the register TV, which's used to get the primary value of LFSR and used to replace the state RS3 with inside the encryption technique. The LFSR is used now no longer only for RS3 updating, however additionally to make sure that the duration of the internal states are as a minimum 216.

3.2 ENCRYPTION/DECRYPTION:

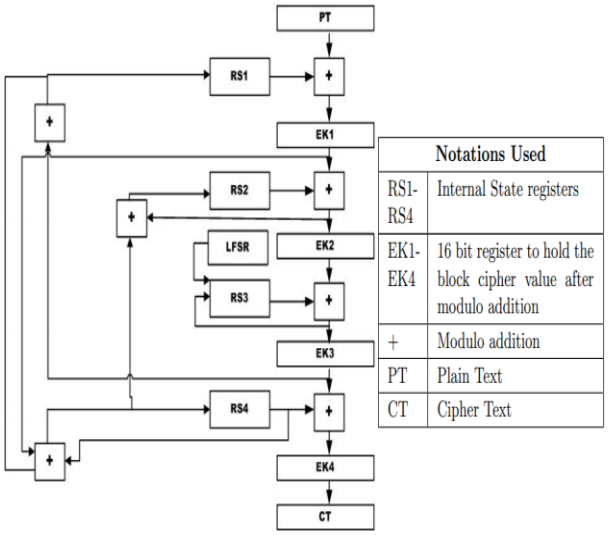
After the initialization technique, encryption begins via taking the input as the Plain Text(PT) observed through the modulo 216 addition (shown in fig.2) of simple text with internal state register, RS1, and the end result is implemented to the block cipher EK1.The complete technique of encryption is shown in Fig.2. This technique is repeated 4 times and produces the cipher-text. When all the 4-block ciphers are completed, the RSi state register is up-to-date accordingly. The decryption technique is simply the opposite operation of the encryption, as shown

ISSN (Online) 2394-2320

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

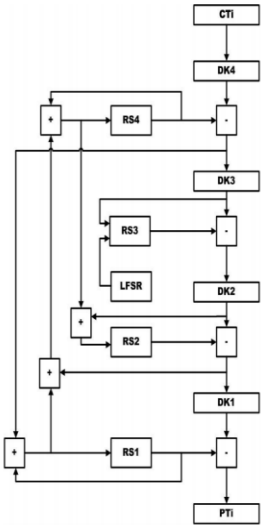
Vol 8, Issue 7, July 2021

in Fig.3.



Notations Used	
RS1-RS4	Internal State registers
EK1-EK4	16 bit register to hold the block cipher value after modulo addition
+	Modulo addition
PT	Plain Text
CT	Cipher Text

Fig. 2 Encryption



Notations Used	
RS1-RS4	Internal State registers
DK1-DK4	16 bit register to hold the value after modulo subtraction
-	Modulo subtraction
+	Modulo addition
PTi	Plain Text
CTi	Cipher Text
LFSR	Linear Feedback Shift Register

Fig. 3 Decryption

3.3 BLOCK CIPHER:

Block cipher utilized in encryption technique as shown in Fig.4. It includes 4 rounds of operation and a very last round. One ordinary round operation includes a key blending step, a substitution step, and a linear transformation step. The sub-key of 64-bit is separated into 4 16-bit round keys in order to be utilized in the subsequent corresponding rounds respectively. In the important thing mixing technique, plaintext block makes use of an exclusive-OR with the round-key. The S-box produces the results step by step. The substitution round makes use of four Serpent-type S-boxes with the input and output of four bits. Table 1 indicates the substitution value of 4bits used for the substitution technique. Here, every four-bit input is substituted with any other four-bit to make confusion in output. The very last step makes use of one replacement step. There is no linear transformation step inside the very last round. The linear transformation step is described in equation (1) wherein an XOR operation is carried out among Din and 6 times right-shifted value of Din and 10 times the right-shifted value of Din.

Dout = Din ^ (Din << 6) ^ (Din << 10) → (1)

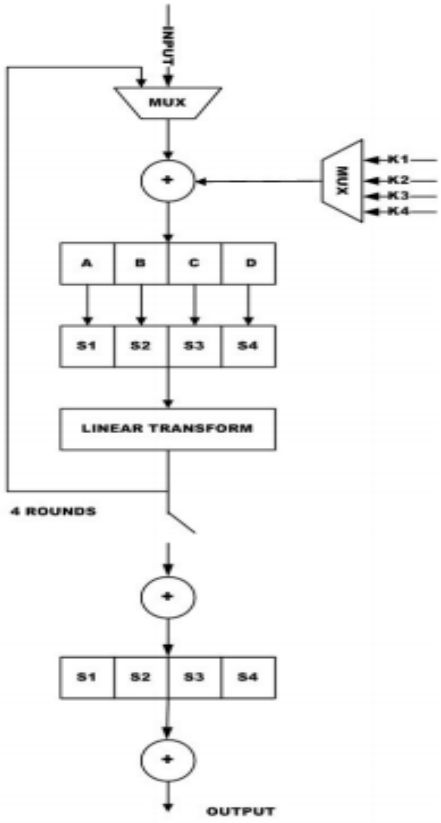


Fig. 4 Block cipher

PROPOSED ARCHITECTURE

Until now, the hummingbird is carried out throughout unique goal platforms. Each of these implementations is both focusing area optimization through looping method or throughput optimizing through loop unrolling of a

ISSN (Online) 2394-2320

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 7, July 2021

block cipher. The proposed block cipher is just like the preceding models, with a change that the cipher loop is partly unrolled. This architecture is a compromise among earlier looped and loop unrolled architecture in order that it attempts to optimize each area and throughput. It requires 6 XORs, 12 S-Boxes, a linear transform, and a pair of multiplexers. Here, simple textual content may be encrypted in eight clock cycles. So the proposed structure is an optimization of area and speed. The 4 rounds of the encryption are unrolled and the 3 operations around substitution, permutation, and linear transformation are carried out in sequence. Since the 4 rounds are carried out in parallel it calls for four times duplication of hardware however at the same time the number of clock cycles required is only 1 and for the final round it calls for one more extra. This encrypted block of 16-bit information is handed for the subsequent five rounds in a looped pipelined way with pipelining registers R1 and R2. The very last encrypted information is taken in a 64-bit register at each clock cycle, so it requires five more clock cycles so in total it calls for 7 clock cycles for getting the encrypted information.

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
s1	8	6	5	f	1	c	a	9	e	b	2	4	7	0	d	3
s2	0	7	e	1	5	b	8	2	3	a	d	6	f	6	2	9
s3	2	e	f	5	c	1	9	a	b	4	6	8	0	7	3	d
s4	0	7	3	4	c	1	a	f	d	e	6	b	2	8	9	5

Table 1 Substitution S box values

Instead of the use of the 4 sbox as such in the design, a hardware-friendly sbox is chosen from the sbox given in Table 1, and it is repeated 4 times to enhance the area and performance. The device is made reset earlier than the primary encryption starts. Upon reset, the control signals data_sel, rs_sel, key_sel, init_encr, and the counters reset to zero. Data_sel is used to choose corresponding input information to the block cipher, rs_sel is used to choose the specified internal register throughout the operation, key_sel is used to choose corresponding subkeys, and init_encr is used to distinguish among initialization and encryption. At first, the internal registers are loaded with the 16-bit random nonce values and the core starts encrypting the RS1+RS3 for 4 iterations. Each iteration takes 7 clock cycles in addition to internal state updating and the identical block cipher architecture is reused every time which guarantees most architecture reuse and therefore better area and performance. After 4 iterations, the initialization is finished and the control signal init_encr is set to 1. Once the initialization is finished, i^{th} simple textual content is taken because of the input, and after 7 clock cycles, we get i^{th} ciphertext. During the above procedure, the corresponding subkeys, internal register, and input information are decided on in line with the control signals key_sel, rs_sel, and data_sel respectively. The control signals are generated primarily based totally on counters. Control signals rs_sel and data_sel must be up-to-date after every block cipher operation, so it is managed through a block counter. The init_encr signal is up-to-date after the initialization technique, so it is managed through a block counter. After every new release, the internal states are up-to-date accordingly. The LFSR is initialized after the initialization technique and up to date after every encryption. The proposed architecture isn't changing the algorithm; it is an optimized hardware implementation of the identical.

RESULTS AND DISCUSSION

The architecture for encryption core in addition to decryption core is modeled using Verilog HDL and simulated in ModelSim. The architecture is synthesized and applied using the Cadence RTL compiler and encounter.

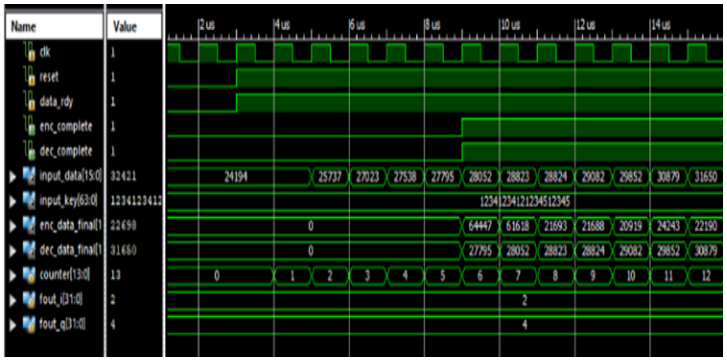


Fig. 5 Simulation result of encryption and decryption

These encrypted and decrypted data pixels are used to form the image using Matlab software.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 7, July 2021

Fig. 6 The input image, encrypted image and decrypted image

Fig. 6 shows that the input image and decrypted image are similar and the encrypted image is different which shows that the simulation is correct.

CONCLUSION

This paper presented an efficient VLSI architecture for an ultra-light-weight cryptographic algorithm named hummingbird. The preceding implementation of this algorithm became specializing in optimizing both the throughput and area. But this paper presented a layout this is optimized in terms of each inside the area in addition to throughput. The proposed design is a compromise among each area optimized and throughput optimized schemes. The hummingbird algorithm has proper performance as compared to all different algorithms, and additionally, it has the smallest block size as compared to all. The experimental outcomes display that the hummingbird algorithm could be very lots beneficial for resource-restricted embedded devices. The initialization of internal registers is executed by loading a few random nonce values which highly influence the security of the algorithm. The technology of those random values at the hardware is executed in the usage of an LFSR.

REFERENCES

1. D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B. S Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim and S. Chee. (2006), "HIGHT: a new block cipher suitable for Low-Resource device", Proceedings of CHES 2006, volume 4249 of LNCS, pages 46-9, Springer.
2. F. X Standaert, G. Piret, N. Gershenfeld and J.-J. Quisquater. (2006), "SEA: a scalable encryption algorithm for small embedded application", Smart Card Research and Applications, Proceedings of CARDIS 2006, volume 3928 of LNCS, pages 222-236, Springer-Verlag.
3. G.Leander, C.Paar, A. Poschmann and K. Schramm. (2007), "New LightweightDES Variants", Fast Software Encryption.
4. A.Poschmann, A.bogdanov and L.R Knudsen. (2007), PRESENT: An UltraLightweight Block Cipher, springer.
5. D. Engels, X. Fan, G. Gong, H. Hu and E. M. Smith. (2010), "Hummingbird:Ultra-Lightweight Cryptography for Resource- Constrained Devices", toappear in the Proceedings of The 14th International Conference on Financial Cryptography and Data Security - FC 2010, Berlin, Germany:Springer-Verlag.
6. X.Fan, G. Gong, K.Lauffenburger and T.Hicks. (2010), Design Space Exploration of Hummingbird Implementations on FPGAs, Techincal Report.
7. Xinxin Fan, Honggang Hu, Guang Gong1, Eric M. Smith and Daniel Engels. (2009), "Lightweight Implementation of Hummingbird Cryptographic Algorithm on 4-Bit Microcontrollers", Institute of Electrical and Electronics Engineers.
8. Xinxin Fan, Guang Gong, Ken Lauffenburger and Troy Hicks. (2010), "FPGA Implementations of the Hummingbird Cryptographic Algorithm", 978- 1-4244-7812-5/10/, IEEE. Advances in Systems Science and Application (2015) Vol.15 No.4 365
9. Biao Min, Ray C.C. Cheung and Yan Han. (2011), "FPGA-based HighThroughput and Area-Efficient Architectures of the Hummingbird Cryptography", 978-1-61284-972-0/11/, IEEE.
10. Ismail San and Nuray At. (2011), "Compact Hardware Architecture for Hummingbird Cryptographic Algorithm", 21st International Conference on Field Programmable Logic and Applications, 978-0-7695-4529-5/11, IEEE.
11. J.-P. Kaps. (2008), "Chai-tea, cryptographic hardware implementations of xTEA", INDOCRYPT 2008, LNCS, vol.5365, pp.363-375, Springer.
12. P. Yalla and J.P. Kaps. (2009), "Lightweight Cryptography for FPGAs", International Conference on Re-ConfigurableComputing and FPGAs ReConFig'09.
13. F. Mace, F.X. Standaert and J.J. Quisquater. (2007), "FPGAimplementation(s) of a Scalable Encryption Algorithm", IEEETrans, Very Large Scale Integ, (VLSI) Syst.Vol.16, No.2, pp.212-216.