1. Manoj. G[1], 2.Roopa Jayasingh.J [2], 3. Divya. P.S[3], 4.Saravanan [4]

Assistant Professor, Department of ECE, Karunya Institute of Technology and Sciences, Coimbatore, India (1,2), Assistant Professor, Department of Mathematics, Karunya Institute of Technology and Sciences, Coimbatore, India (3), Professor, Department of ECE, P.S.R Engineering College (4)
ORCID: 1. 0000-0003-3543-5672; 3. 0000-0002-3836-2432

doi:10.15199/48.2023.06.05

Comparative analysis of low power implementation for AES algorithm in ARTIX 7 FPGA & ASIC

Abstract. *Encryption is a mandate in today's information sharing based society. Various Algorithms have been proposed and used to implement encryption. The AES algorithm is one such encryption algorithm widely known for its faster encryption speeds and withstanding ability against cyber-attacks. Its resilience comes from the fact that it can use 128 or 192- or 256-bit keys to encrypt 128, 192 or 256 bit plain text. The AES algorithm has been implemented in ASIC and FPGA to realize the best practices for the implementation of the algorithm for efficient usage. The power, area and timing analysis from both implementations have been compared to infer the best implementation strategy. The experimental results indicate that care has to be taken to reduce switching activity of signals which were observed to be the primary contributor of dynamic power consumption. Recommendations have been included to reduce signal switching power consumption during Logic BIST designs for the algorithm. The power analysis show that ASIC implementation of the AES algorithm would be much more beneficial in comparison to ARTIX 7 FPGA implementation.*

Streszczenie. *Szyfrowanie jest obowiązkiem w dzisiejszym społeczeństwie opartym na wymianie informacji. Zaproponowano i wykorzystano różne algorytmy do implementacji szyfrowania. Algorytm AES jest jednym z takich algorytmów szyfrowania, powszechnie znanym z większej szybkości szyfrowania i odporności na cyberataki. Jego odporność wynika z faktu, że może używać kluczy 128-, 192- lub 256-bitowych do szyfrowania zwykłego tekstu 128, 192 lub 256-bitowego. Algorytm AES został zaimplementowany w ASIC i FPGA, aby zrealizować najlepsze praktyki implementacji algorytmu w celu efektywnego wykorzystania. Porównano analizę mocy, obszaru i czasu z obu wdrożeń, aby wywnioskować najlepszą strategię wdrożenia. Wyniki eksperymentów wskazują, że należy zwrócić uwagę na zmniejszenie aktywności przełączania sygnałów, które były głównymi sprawcami dynamicznego poboru mocy. Uwzględniono zalecenia dotyczące zmniejszenia poboru mocy przy przełączaniu sygnału podczas projektowania logiki BIST dla algorytmu. Analiza mocy wykazała, że implementacja ASIC algorytmu AES byłaby dużo bardziej korzystna w porównaniu z implementacją ARTIX 7 FPGA* (***Analiza porównawcza implementacji małej mocy dla algorytmu AES w ARTIX 7 FPGA & ASI***C)

Keywords: Field Programmable Gate Array (FPGA), Application Specific Integrated Circuit(ASIC), Advanced Encryption Standard (AES),
Słowa kluczowe: FPGA, SPC – specified Integrated Circuit

Introduction

Data can be scrambled using encryption so that only authorised parties can decipher it. Technically speaking, it is the process of changing plaintext that can be read by humans into cypher text, which is incomprehensible text. In plainer terms, encryption [1] changes readable data to make it seem random. A cryptographic key, or collection of numbers that the sender and the recipient of an encrypted message both agree upon, is needed for encryption. Despite the fact that encrypted[2] data appears random, encryption works in a logical, predictable manner, making it possible for someone who gets encrypted data and has the proper key to decrypt it and restore it to plaintext. A third party would be extremely unlikely to be able to decrypt or crack the cypher text using brute force when using keys for truly secure encryption.

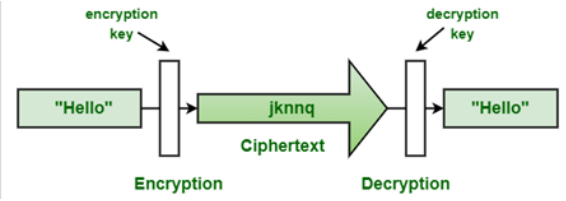

Fig.1. Block diagram of Cryptography

Types Of Encryption

Symmetric and asymmetric encryption are the two major types of encryption. Public key encryption[3] is another name for asymmetric encryption. Symmetric encryption uses a single key that is used by all communicating parties for both encryption and decryption. Asymmetric encryption, often known as public key encryption, uses two keys: one is used for encryption and the other for decryption. The encryption key[4][5] is shared openly for use by anybody, however the decryption key is kept secret (thus the term "private key") (hence the "public key" name). The block diagram of the cryptography is shownin the figure 1 as given below.

Encryption Algorithms

Data are converted into cypher text using an encryption algorithm. The data will be altered by an algorithm using the encryption key in a predictable fashion, such that even though the encrypted data will seem random, it can be decrypted and returned to plaintext [6] with the decryption key.

Typical symmetric encryption techniques are as follows:

- AES(Advanced Encryption Standard)
- 3-DES(Data Encryption Standard)
- SNOW

Typical assymmetric encryption techniques are as follows:

- RSA
- Elliptic curve cryptography

A brute force assault is when an attacker who is unaware of the decryption key [7] makes millions or billions of guesses in an effort to get the key. The majority of contemporary encryption techniques are resistant to brute force assaults when used with strong passwords.

AES

For AES-128/198/256 bit, the full procedure [8] takes 10/12/14 rounds, accordingly. The input data and key for AES-128 bit are both 128 bits, and each round takes one cycle to complete. The architectural flow [9] of AES is displayed here.

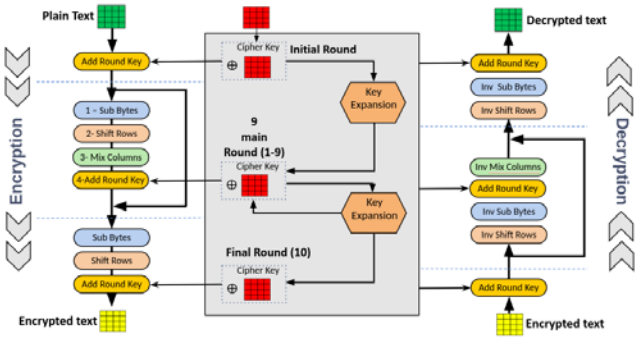

Fig.2. AES Flow Architecture

Internally, the AES algorithm's operations are performed on a two-dimensional array of bytes called the State. So, at the beginning of the Cipher or Inverse Cipher, the input array,

'in', is copied to the State array according to the scheme. The figure 2 gives the AES architecture flow of the system.

$$s[r, c] = in[r + 4c].$$

(1)

The State array's four bytes are organised into four 32-bit words in each column, with the row number r serving as an index for each word's four bytes. In light of this, the state can be described as a one-dimensional [11] collection of 32-bit words (columns), w0–w3, where column number c acts as an index. State can be thought of as an assortment of the following four words:

$$w0 = s0,0\ s1,0\ s2,0\ s3,0\ w2 = s0,2\ s1,2\ s2,2\ s3$$

(2) $$w1 = s0.1\ s1.1\ s2.1\ s3.1\ w3 = s0.3\ s1.3\ s2.3\ s3$$

Each round of AES algorithm contain few steps shown below (except round 10):

- Add round key
- Substitute bytes
- Shift rows
- Mix columns

ADD round key

A Round Key is added to the State in the AddRoundKey transformation using an implemented bitwise XOR operation. This is the AES algorithm's initial phase, and it is Only an XOR operations. The Add Round off for the AES[12] for a sample is shown in the figure 3

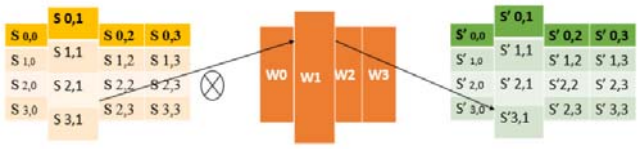

Fig.3. AES Add Round off

ADD Round Key Operation

The matrix of 16 bytes are consider as 128 bits and exored to 128 bits of the round key[13]. If last round is this then output is 128 bits Encrypted output. Otherwise, these 128 bits will again go to the similar round considering 16 bytes.

SUB-BYTES

Byte is changed to a value in the S-box through a non-linear transformation. The S-box is already set up to be used in the algorithm. Data substitution[14] is done with the S-box. S-box can be thought of simply as a lookup table. Each block has 8 bits of data, and the first 4 bits can be seen as a row index and the last 4 bits as a column index[15][16]. By using these row and column indexes, we can retrieve the value from the S-box is shown in the figure 4,5.

It is a non-linear transformation where byte is replaced with a value in S-box. The S-box is predetermined for using it in the algorithm. S-box is used to substitute data. Simply we can see S-box as a lookup table. The way to substitute

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	b2	c9	7d	fa	59	47	f0	ad	d4	s2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Fig.5. S-box matrix for the AES Operation

SHIFT rows

In this operation, each row of the state is cyclically shifted to the left, depending on the row index. The 1st row is shifted 0 positions to the left. The 2nd row is shifted 1 position to the left. The 3rd row is shifted 2 positions to the left. The 4th row is shifted 3 positions to the left. The shift operation of the AES is shown in the figure 6.

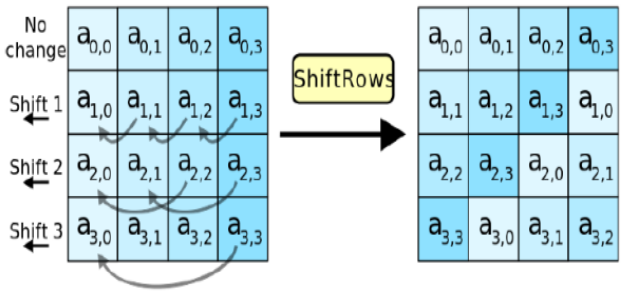

Fig.6. Shift Row Operation

MIX columns

The Mix Columns transformation operates on the State column-by-column, treating each column as a four-term polynomial. The columns are considered as polynomials over GF (28) and multiplied modulo x4 + 1 with a fixed polynomial. An illustration is shown in the figure 7 to obtain the mix column.

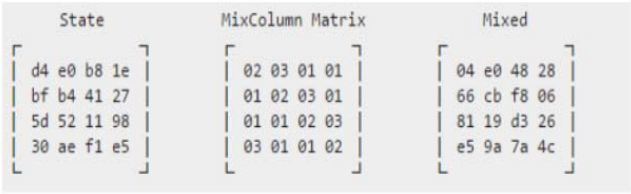

Fig.7. An example of the figure to provide the mix column

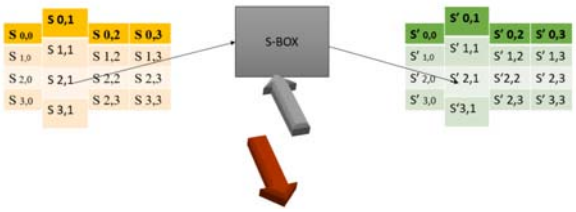

Fig. 8. Substitute Bytes Operation

KEY EXPANSION

Plain / Cipher Text. And next round onward Expanded Key from Expanded Key Schedule is XORed with dataig.

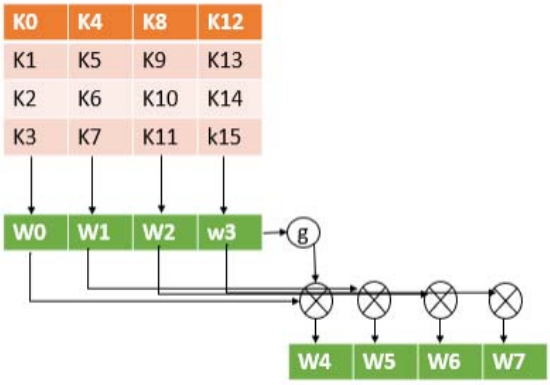

Fig. 9. Key Expansion column logical Circuit

The key expansion of the logical circuit is shown in the figure 7 which provides the logic for then key expansion for the cypher text.

Inference Result and Analysis

The AES algorithm was programmed using the Hardware Description Language (HDL) Verilog and simulated using the CADENCE NCLaunch eda tool. Figure 8 is the simulation output taken from the tool.

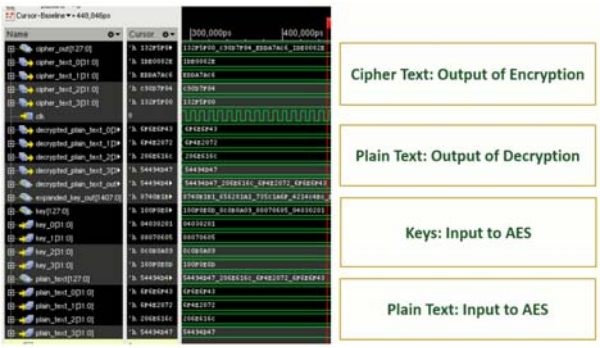

Fig. 10. Simulation result of the cipher text of the Encrypted logic

After encryption the algorithm outputs the 128 bit cipher text 32F500_C90D784_EDDA7AC6_1DE0082E. The cipher text is displayed on the top portion of the figure. AES receives the cipher text as input during decryption and outputs the corresponding 128 bit long plain text. The decrypted plain text is displayed as the second portion of the figure 8.

The AES key expansion algorithm takes as input a 4-word key and produces a linear array of 44 words. Each round uses 4 of these words each word contains 32 bytes which means each sub-key is 128 bits long. In first round user key is XORed with the original

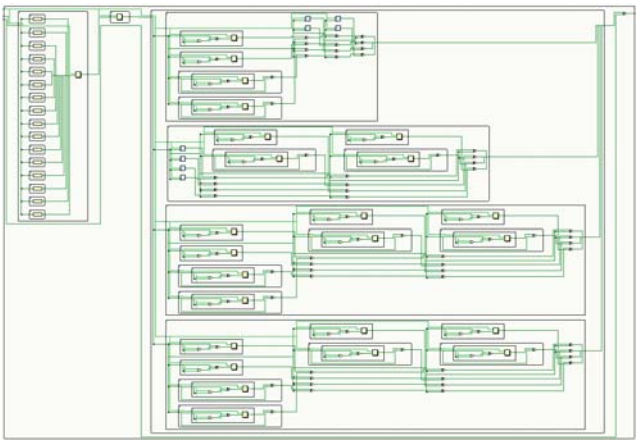

Fig. 11 An Implementation of PGA

The implementation of Xilinx Artix-7 FPGA AC701 FPGA of the logical circuit is shown in the figure 7 which provides the logic for then key expansion for the cypher text. The resource utilization of the Xilinx Artix-7 FPGA AC701 is shown in the tabulation 1with the percentage of the resource.

Table 1: Utilization of the Xilinx Artix-7 FPGA AC701

Resource	Utilization	Available	Utilization %
LUT	9694	133800	7.25
FF	3712	267600	1.39
IO	385	400	96.25
BUFG	1	32	3.13

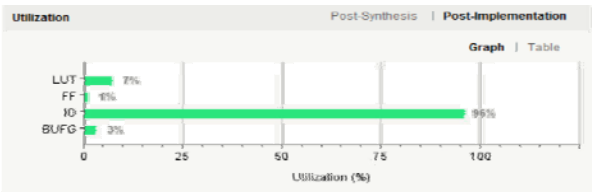

Fig.12. An implementation of FPGA

The AES algorithm was programmed using the Hardware Description Language (HDL) Verilog and simulated using the CADENCE encounter, Figure 11 is the layout model of ASIC output taken from the tool using 90 nm technology.

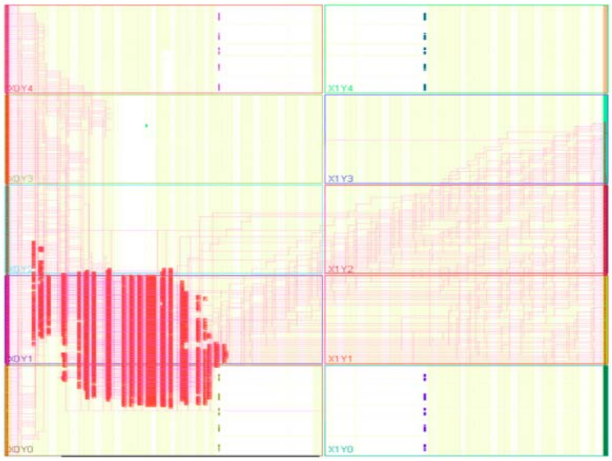

Fig.13. LAYOUT MODEL FOR ASIC using 90 nm Technology

Power analysis of the FPGA and cadence is done for the AES system and the comparison is shown in the figure 12. The anlaysis shows that the power consumpsion is very less and can be used in low power applications.

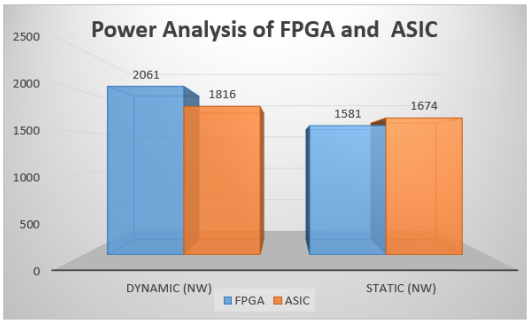

Fig.14. An implementation of FPGA

Conclusion

The ASIC and FPGA implementations of the Advanced Encryption Algorithm for 128 bit encryption and decryption. It has been observed through power analysis that in both cases of implementation that the dynamic power consumption far outweighs the static and leakage power dissipation. Signal transitions were found to be the primary contributor of dynamic power consumption. The inference from this observation is that in order to design Low power AES systems, care has to be taken to reduce the number of signal transitions. Comparing the area analysis of the FPGA and ASIC implementations, it could be seen that ASIC implementation of the AES algorithm would be a much useful approach over FPGAs. This conclusion is arrived from the observation that the IO requirements of the AES algorithm even for a 128 bit encryption is significant and would make an FPGA almost unavailable for any other purposes.

***Authors**: Dr. Manoj G received his B. Tech degree from Amritha University, Coimbatore and M.Tech degree from Karunya University, Coimbatore, India, Ph.D from Anna University.The area of research include VLSI, AI, IoT SoC.*

Roopa Jayasingh J. completed her UG and PG in Karunya Technology and Sciences. She completed her PhD in Information and Communication Engineering under the Anna University, Coimbatorefocusing on biomedical signal processing. She is currently working as an Assistant Professor in Electronics and Communication Engineering, Karunya Institute of Technology and Sciences, Coimbatore. Her Research interest is in medical image processing, deep learning and internet of things, etc.

Dr.P.S. Divya is working as Assistant Professor in Karunya Institue of technology and Sciences, Coimbatore. The area of research include Mathematical Modelling of System, Renewable Energy optimization, IoT, AI for real time application.
Corresponding Email: divya_deepam@karunya.edu

Dr. K. Saravanan, Professor ECE (NBA Accredited), P.S.R Engineering College,Accredited by NAAC A+. The area of reserach include Signal Modelling of System,VLSI optimization, IoT, AI for real time application.

REFERENCES

[1] L. Bossuet, M. Grand, L. Gaspar, V. Fischer, and G. Gogniat, “Architectures of flexible symmetric key crypto engines—a survey: From hardware coprocessor to multi-crypto-processor system on chip,” ACM Comput. Surv.,(2013), vol. 45.

[2] Johnson B., Pike G.E., Preparation of Papers for Transactions, *IEEE Trans. Magn*., 50 (2002), No. 5, 133-137

[3] Aiwu Ruan , Shi Kang, Yu Wang, Xiao Han, Zujian Zhu, Yongbo Liao, Peng Li. A,Built-In Self-Test (BIST) system with non intrusive TPG and ORA for FPGA test and diagnosis. *Microelectronics Reliability, (*2013) Volume 53.

[4] Elham Moghaddam , Nilanjan Mukherjee, Janusz Rajski, Jedrzej Solecki, Jerzy Tyszer,and Justyna Zawada, Member Logic BIST With Capture-Per-Clock Hybrid Test Points.*IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS*, VOL. 38 - 2019

[5] Gert Jervan, Elmet Orasson, Helena Kruus, Raimund Ubar. Hybrid BIST optimization using reseeding and test set compaction. *Microprocessors and Microsystems* Volume 32-2008

[6] Gundolf Kiefer, Harald Vranken, Erik Jan Marinissen & Hans-Joachim Wunderlich Application of Deterministic Logic BIST on Industrial Circuits. *Journal of Electronic Testing* volume 17 – 2001

[7] G Sathesh Kumar et al. Fuzzy Logic based Truly Random Number Generator for high speed BIST appilications. Microprocessors and Microsystems, 2019

[8] Michael Filipek, Grzegorz mrugalski et al. Low Power Programmable PRPG With Test Compression Capabilities. IEEE transactions on Very large Scale Integration Systems Volume 23- 2015

[9] Y. Cao, Predictive technology model for robust nanoelectronic design.Springer Science & Business Media, 2011.

[11] M. R. Guthaus, J. E. Stine, S. Ataei, B. Chen, B. Wu, and M. Sarwar,“OpenRAM: An open-source memory compiler,” in ICCAD. IEEE, 2016, pp. 1–6.

[12] Aneesh, K., Manoj, G., Shylu Sam, S Design Approaches of Ultra-Low Power SAR ADC for Biomedical Systems - A Review, Journal of Circuits, Systems and Computers.(2022),223-239.

[12] D. Reis, K. Ni, W. Chakraborty, X. Yin, M. Trentzsch, S. D. Dunkel, ¨T. Melde, J. Muller, S. Beyer, S. Datta, M. T. Niemier, and X. S. Hu, ¨“Design and analysis of an ultra-dense, low-leakage, and fast fefet-based random access memory array,” IEEE Journal on Exploratory Solid-State Computational Devices and Circuits, vol. 5, pp. 103–112, 2019.

[13]Mutlu, “Memory scaling: A systems architecture perspective,” in 2013 5th IEEE International Memory Workshop, 2013, pp. 21–25.

[14] Rohit Kapur, Srinivas Patil, Thomas J. Snethen, and T. W. WilliamsA Weighted Random Pattern Test Generation System. IEEE TRANSACTIONS ON COMPUTERAIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, VOL 15 – 1996.

[15] Sameh El-Ashry et al. On Error Injection for NoC Platforms: A UVM-based Generic Verification Environment. IEEE Transactions on Computer-Aided Design of Integrate Circuits and Systems Volume 39 – 2020

[16]Vishnupriya Shivakumar, C. Senthilpari, Zubaida Yusoff. Test power and area optimized logic built-in self-test with higher fault coverage for automobile SoCs.Microelectronics Journal Volume 124- 2022.