A Compact, Low Power AES Core on 180nm CMOS Process

Van-Lan Dao, Van-Phuc Hoang, Anh-Thai Nguyen and Quy-Minh Le
Le Quy Don Technical University, 236 Hoang Quoc Viet Str., Hanoi, Vietnam
Email: kqha1025@gmail.com; phuchv@mta.edu.vn; nguyenanhthai77@gmail.com; minhmhk@gmail.com

***Abstract*— This paper presents a compact, low power AES cryptography core with a small S-box and an improved key expansion block for emerging wireless networks. The implementation results with an 180nm CMOS standard library show that the proposed AES core can reduce the area and power consumption significantly.**

Keywords— AES; ASIC; low power; low area

I. INTRODUCTION

Currently, wireless networks are highly employed for many applications such as personal area connection, broadband internet connection, smart home, smart environment monitoring, etc. [1, 2]. Wireless sensor network is one of emerging wireless networks for smart society applications. Due to the employment of the wireless channel, secure connectivity is becoming a more and more essential issue for these networks [1]. Advanced Encryption Standard (AES) is a well-known security standard for data encryption and decryption [3, 4]. Although the AES algorithm has been standardized, the efficient hardware architecture and implementation methods are the topics which many researchers are focusing on. However, with the fast development of many portable, wearable applications and devices, especially the Internet of things (IoT), the low area, low power and secure hardware implementations are highly required. Therefore, the higher power efficiency VLSI impelmentations are highly expected. In the era of IoT, low power and high security requirements can be promisingly fulfilled by hardware cryptography implementation.

The objective of this paper is to design a compact, low power AES core which includes both encryption and decryption functions for such area and power constrained wireless networks and applications. Our main contribution is that a compact, low power AES core implementation is proposed by combining several optimized components in the AES core and some modification in 8-bit AES architecture for high hardware resource efficiency in the ASIC platform. In this paper, Section II describes the compact AES core architectures. Section III and section IV present the optimized S-Box and improved key-expansion unit which are two essential components in AES cryptography cores. Then, section V presents the implementation results and section VI concludes the paper.

II. COMPACT AES CORE ARCHITECTURES

AES encryption core processes data in 128-bit blocks with the key lengths of 128, 192 or 256 bits. Figure 1 shows the 128-bit AES encryption/decryption algorithms. The left hand side is the encryption flow and the right hand side is the decryption one. In this paper, to reduce the AES encryption core area, we employ 8-bit architecture with compact S-boxes so that the AES core encrypts one 8-bit data block in each clock cycle. Authors in [5]-[7] also focused on optimizing AES encryption core for the low area implementation. However, they used an LUT-based (non-optimized) S-box that may result in a high area ASIC implementation. Hence, some papers such as [8]-[14] proposed the optimized S-Box designs for low area AES implementations.

In this paper, firstly, the 8-bit AES core architecture is used for the compact implementation as shown in Figures 2-4. The 8-bit architecture was also employed in [7]. However, the non-optimized S-box leads to more optimization required. In Fig. 2, the AES encryption core includes a key expansion unit, a mix-column unit, a parallel to serial converter and a byte permutation unit.

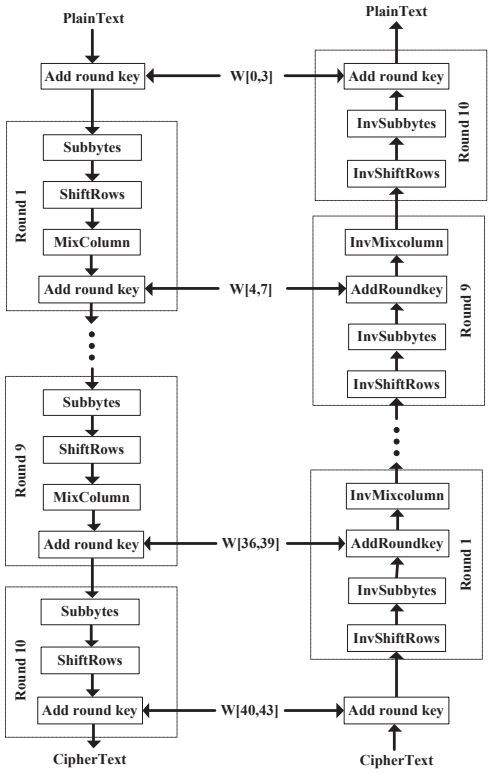


Fig. 1. Standardized AES encryption and decryption algorithms.

Table I lists the function of each signal in the proposed AES core. S-box 1 and S-box 2 are two sub-blocks in the byte permutation unit as described in [14]. The detail implementation of this byte permutation unit will be presented in the next section. In the decryption core as depicted in Fig. 3, an additional inverse S-box is used.

Hence, the overall architecture of the AES encryption/ decryption core is proposed in Fig. 4 by combining the encryption and decryption parts. The 8-bit AES core requires 160 clock cycles for each encryption/decryption operation.

TABLE I. SIGNALS IN THE PROPOSED AES CORE.

Signal	Direction	Description
clk	Input	System clock
rst_n	Input	System reset
load_in	Input	Control signal to load data and key
unload_in	Input	Control signal to unload data and key
start_in	Input	Control signal to start the encryption
inv_in	Input	To select encryption/decryption operation
key_in	Input	Key input
data_in	Input	Data input
data_out	Output	Data output
busy_out	Output	To indicate that the output is ready to read
comp	Output	To indicate that the output is ready to read and the new input data can be fed

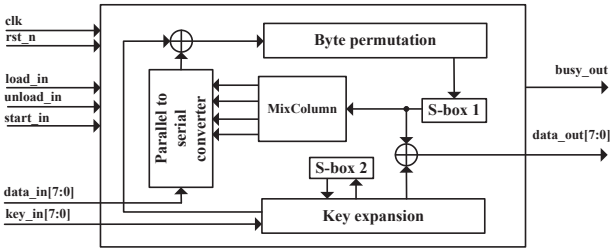


Fig. 2. The 8-bit AES encryption core architecture.

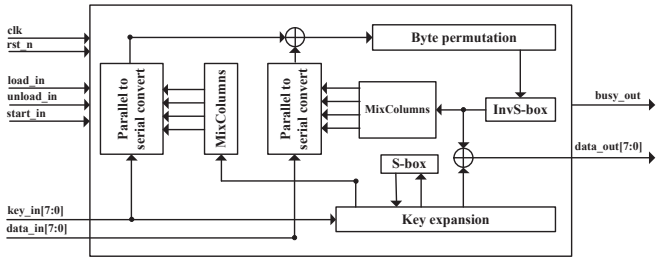


Fig. 3. The 8-bit AES decryption core architecture.

III. S-BOX DESIGN

S-box is an important block in the AES core so that some papers on S-box optimization for the specific requirements have been published [8-14]. It can be optimized for speed or area depending on the application requiring the core. When using the LUT-based architecture, a 256-byte memory block is required so that the area may be high. Therefore, to reduce the complexity, we try to propose an alternative S-box architecture for the compact AES implementation.

Actually, S-box is an 8×8 matrix for the two following transformations. The first one is the byte inversion in which each byte is substituted by its inverted version (by the the multiplication in GF(2^8)) and the second transformation the affine transformation in GF(2^8) according to (1).

$$y_i = x_i \oplus x_{(i+4) \bmod 8} \oplus x_{(i+5) \bmod 8} \oplus x_{(i+6) \bmod 8} \oplus x_{(i+7) \bmod 8} \oplus c_i \qquad (1)$$

in which, $0 \le i < 8$ and x = "$x_0x_1x_2x_3x_4x_5x_6x_7$" is the result of byte inverting, and y = "$y_0y_1y_2y_3y_4y_5y_6y_7$" is the result of affine transformation. Byte c is the constant of {63} or {01100011}. The matrix form of this transformation is shown in (2).

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 10001111 \\ 11000111 \\ 11100011 \\ 11110001 \\ 11111000 \\ 01111100 \\ 00011111 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \qquad (2)$$

As we can see, each bit of one byte in GF(2^8) can be considered as a coefficient for an exponent in the polynomial of GF(2^8). As stated in [11], every component in GF(2^8) can be presented as a linear polynomial with the coefficients in GF(2^4). The linear polynomial can be written in the form of $bx+c$, via a second order polynomial of x^2+Ax+B.

Then, the inverting of any polynomial in the form of $bx + c$ can be shown in (3).

$$(bx+c)^{-1} = b(b^2B + bcA + c^2)^{-1}x + (c + bA)(b^2B + bcA + c^2)^{-1} \qquad (3)$$

In this paper, the S-box is designed as presented in Fig. 5 and based on [13]-[14] to derive an efficient implementation. The S-box is transformed from GF(2^8) architecture to GF(2^8)/ GF(2^4)/GF(2^2) architecture. The linear mapping block (*lin. map*) in Fig. 5 converts the basis from GF(2^8) to GF(2^8)/ GF(2^4)/GF(2^2). After some processing steps [14], the result from GF(2^8)/GF(2^4)/GF(2^2) is mapped to GF(2^8).

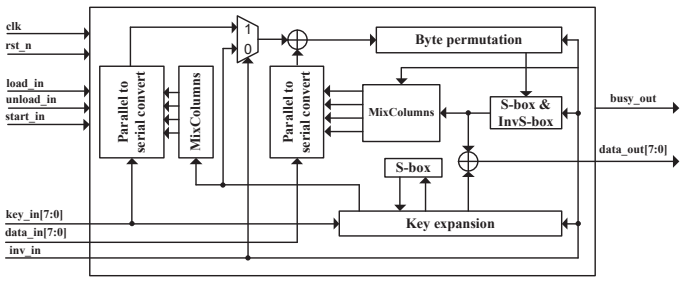


Fig. 4. The 8-bit AES encryption/decryption core architecture.

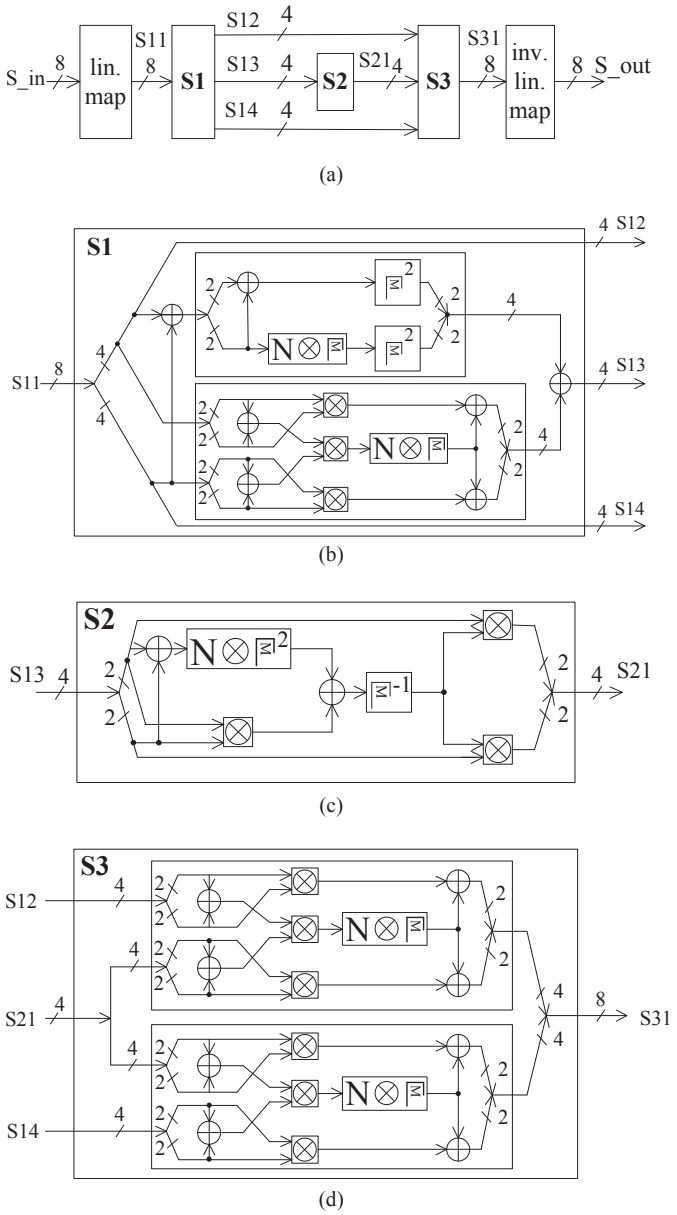


Fig. 5. Compact S-box architecture.

IV. RCON BLOCK OPTIMIZATION FOR KEY-EXPANSION

According to [3], Rcon block which is used in key-expansion takes the input from *r_in* signal. In [7], Rcon is a multiplexer (MUX) circuit which uses *r_in* as the selection signal as shown in Fig. 6a. In our design, Rcon block is optimized by the simple Karnaugh optimization method and the results are presented in (4)-(6) as well as in Fig. 6.

Rcon module description for the cases of encryption and decryption are presented in (4) and (5), respectively. Moreover, for the design with both encryption/decryption, Rcon block is designed as shown in (6) when *inv_in* = '1'. In Fig. 6c, when *inv_in* = '0', Block 2 performs the function which is presented in (4). In contrast, when *inv_in* = '1', Block 1 and Block 2 are identical as presented in (6).

$$\begin{cases} Rcon_7 = r\_in_2.r\_in_1.r\_in_0 \\ Rcon_6 = r\_in_2.r\_in_1.\overline{r\_in_0} \\ Rcon_5 = r\_in_2.r\_in_0.\overline{r\_in_1} + r\_in_3.r\_in_0 \\ Rcon_4 = r\_in_3 + r\_in_2.\overline{r\_in_1}.\overline{r\_in_0} \\ Rcon_3 = \overline{r\_in_2}.\overline{r\_in_0}.r\_in_3 + r\_in_0.\overline{r\_in_2}.r\_in_1 \\ Rcon_2 = \overline{r\_in_2}.\overline{r\_in_0}.r\_in_1 + r\_in_3.r\_in_0 \\ Rcon_1 = r\_in_3 + \overline{r\_in_2}.\overline{r\_in_1}.r\_in_0 \\ Rcon_0 = r\_in_2.\overline{r\_in_1}.\overline{r\_in_0} \end{cases} \quad (4)$$

$$\begin{cases} Rcon_7 = \overline{r\_in_2}.\overline{r\_in_0}.r\_in_1 \\ Rcon_6 = \overline{r\_in_2}.r\_in_1.\overline{r\_in_0} \\ Rcon_5 = \overline{r\_in_3}.\overline{r\_in_1}.\overline{r\_in_0} \\ Rcon_4 = \overline{r\_in_3}.\overline{r\_in_2}.\overline{r\_in_1} + \overline{r\_in_3}.\overline{r\_in_1}.r\_in_0 \\ Rcon_3 = \overline{r\_in_3}.\overline{r\_in_2}.\overline{r\_in_1}.r\_in_0 + r\_in_2.r\_in_1.\overline{r\_in_0} \\ Rcon_2 = \overline{r\_in_3}.\overline{r\_in_2}.\overline{r\_in_1}.\overline{r\_in_0} + r\_in_2.r\_in_1.r\_in_0 \\ Rcon_1 = \overline{r\_in_2}.\overline{r\_in_1}.\overline{r\_in_3} + \overline{r\_in_0} \\ Rcon_0 = \overline{r\_in_2}.\overline{r\_in_1}.\ r\_in_0 \end{cases} \quad (5)$$

$$\begin{cases} temp_3 = \overline{r\_in_3}.\overline{r\_in_2}.\overline{r\_in_1} \\ temp_2 = r\_in_2.\overline{r\_in_1} + r\_in_1.\overline{r\_in_2} \\ temp_1 = r\_in_1 \\ temp_0 = \overline{r\_in_0} \\ Rcon_7 = temp_2.temp_1.temp_0 \\ Rcon_6 = temp_2.temp_1.\overline{temp_0} \\ Rcon_5 = temp_2.temp_0.\overline{temp_1} + temp_3.temp_0 \\ Rcon_4 = temp_3 + temp_2.\overline{temp_1}.\overline{temp_0} \\ Rcon_3 = \overline{temp_2}.\overline{temp_0}.temp_3 + temp_0.\overline{temp_2}.temp_1 \\ Rcon_2 = \overline{temp_2}.\overline{temp_0}.temp_1 + temp_3.temp_0 \\ Rcon_1 = temp_3 + \overline{temp_2}.\overline{temp_1}.temp_0 \\ Rcon_0 = \overline{temp_2}.\overline{temp_1}.\overline{temp_0} \end{cases} \quad (6)$$

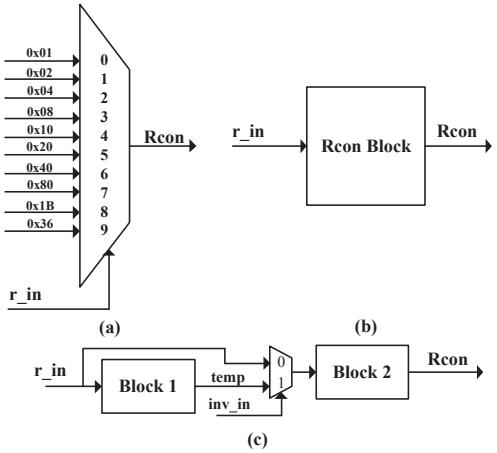


Fig. 6. Rcon block design using a MUX in [7] (a) and using Karnaugh optimization in this paper (b, c).

V. IMPLEMENTATION RESULTS

The 8-bit AES core was implemented with VHDL, simulation in Modelsim tool and then implemented with an 180nm CMOS standard library by Synopsys design tools. Figure 7 is the simulation model for the 8-bit AES encryption core. The input generation block generates the input vector

values for AES core verification. Figure 8 and Fig. 9 present the functional simulation results in Modelsim tool and post-synthesis simulation results in Synopsys VCS tool, respectively. Table II is an example of a test vector for the AES core verification in the case of the encryption operation. The implementation results are presented in Table III and Table IV in which the proposed AES core is compared with the designs in [7], [11], [15], [17]-[20]. These tables have some blank cells because in some other papers, only AES encryption is implemented. It can be seen that the proposed AES core can reduce the area and power consumption significantly compared with some other designs. Figure 10 is the layout of proposed 8-bit AES core with an 180nm CMOS standard cell library.

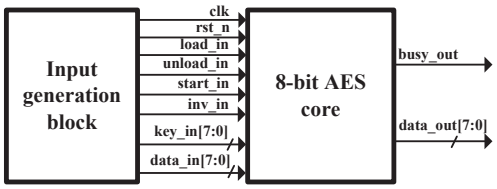


Fig. 7. The simulation model for the 8-bit AES core.

TABLE II. A TEST VECTOR FOR AES CORE VERIFICATION.

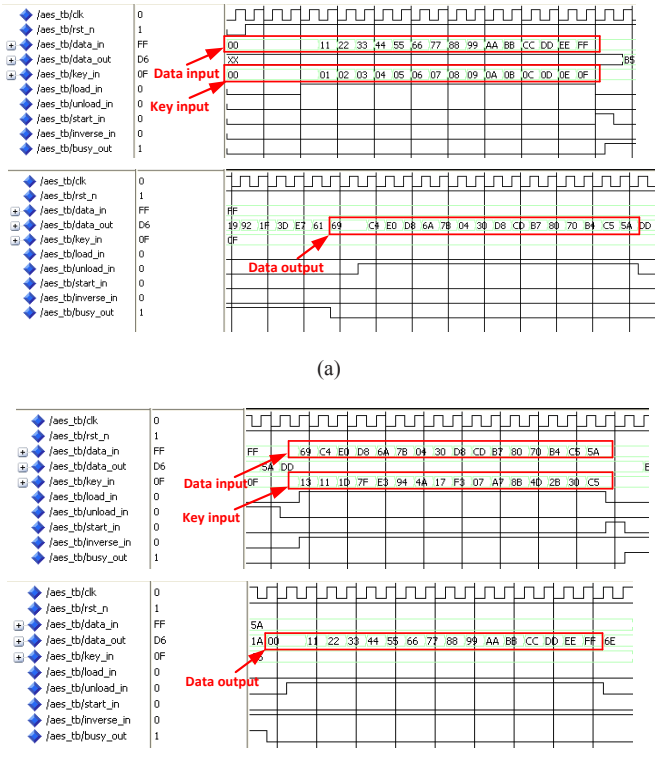
data_in (hexa)	key_in (hexa)	data_out (hexa)
0x00,0x11,0x22,0x33, 0x44,0x55,0x66,0x77, 0x88,0x99,0xAA,0xBB, 0xCC,0xDD,0xEE,0xFF	0x00,0x01,0x02, 0x03,0x04,0x05, 0x06,0x07,0x08, 0x09,0x0A,0x0B, 0x0C,0x0D, 0x0E,0x0F	0x69,0xC4,0xE0,0xD8, 0x6A,0x7B,0x04,0x30, 0xD8,0xCD,0xB7,0x80, 0x70,0xB4,0xC5,0x5A

TABLE III. IMPLEMENTATION RESULTS OF PROPOSED AES CORE IN 180NM CMOS TECHNOLOGY, COMPARED WITH OTHER DESIGNS.

Paper	Encrypt		Decrypt		Encrypt/Decrypt	
	Area	Speed (MHz)	Area	Speed (MHz)	Area	Speed (MHz)
Our work	2.9kgates	50.5	3.7kgates	50.0	4.2kgates	50.5
[7]	3.1kgates	152	-	-	-	-
[15]	2200μm^2	1100	2736μm^2	1100	-	-
[17]	3.4kgates	80	-	-	-	-
[18]	5.5kgates	12	-	-	-	-
[19]	3.5kgates	890KHz @0.4V	-	-	-	-
[20]	0.012 mm^2	11MHz @0.5V	-	-	-	-

TABLE IV. POWER COMSUMPTION ESTIMATION RESULTS OF PROPOSED AES CORE COMPARED WITH OTHER DESIGNS.

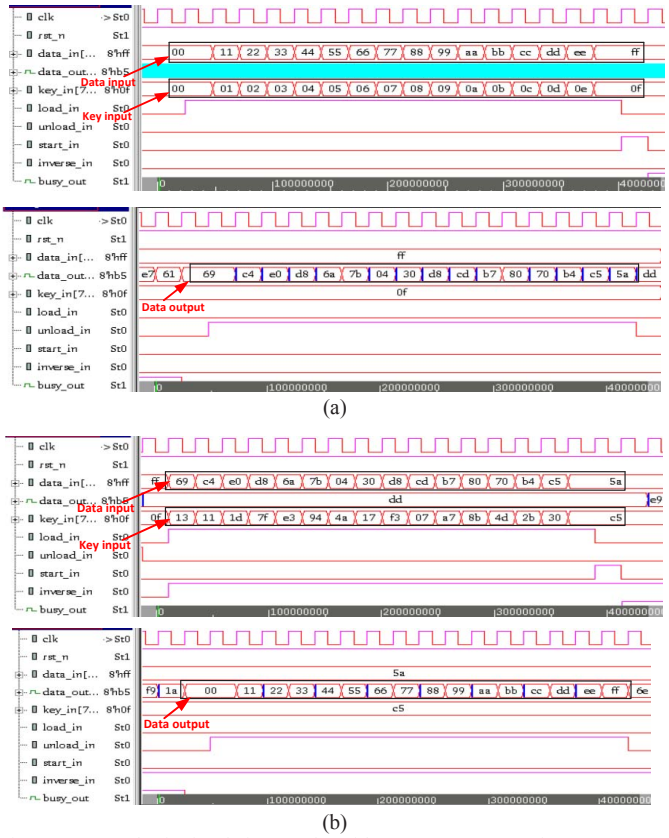
Paper	Encrypt (μW/MHz)	Decrypt (μW/MHz)	Encrypt/Decrypt (μW/MHz)
Our work	34	40	46
[7]	37	-	-
[17]	4.5	-	-
[18]	99	-	-
[19]	0.85-100 kbps/0.4 V	-	-
[20]	14.6 μW @ 0.5V	-	-



(a)

(b)

Fig. 8. Simulation results in Modelsim tool: Encryption (a); Decryption (b).



(a)

(b)

Fig. 9. Post-synthesis simulation results with Synopsys VCS tool: (a) Encryption; (b) Decryption.

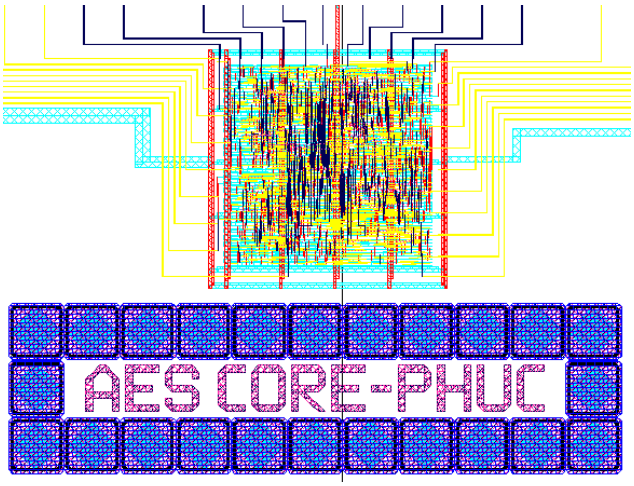


Fig. 10. Layout of the proposed AES core using 8-bit architecture with 180nm CMOS technology, the core layout dimension is 300×300µm.

VI. CONCLUSIONS

This paper has presented a low power, area efficient AES core for emerging wireless networks. The implementation results in an 180nm CMOS ASIC library show that by using an optimized S-box and an improved Rcon design, the AES encryption core area can be reduced to 2.9kgates and power consumption can be reduced to 34µW/MHz. Therefore, this AES cryptography core is highly potential to be used in energy constrained wireless network applications such as wireless sensor networks, IoT systems for environment monitoring which requires both low power consumption and secure compact cryptography cores. In the future, we will further optimize the power consumption for the proposed AES core and apply it for a wireless network application.

ACKNOWLEDGMENT

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.02-2015.20.

REFERENCES

[1] Xiaojiang Du, Hsiao-Hwa Chen, "Security in wireless sensor networks," *IEEE Wireless Communications*, vol.15, no.4, pp.60-66, Aug. 2008.

[2] Wei Wang, Guangyu He, Junli Wan, "Research on Zigbee wireless communication technology," *Pro. 2011 International Conference on Electrical and Control Engineering (ICECE)*, pp.1245-1249, Sep. 2011.

[3] National Institute of Standards and Technology (NIST), "Advanced Encryption Standard (AES)," *FIPS Publication 197*, Nov. 2001.

[4] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Inc. Boca Raton, FL, USA, 1996.

[5] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A compact Rijndael hardware architecture with S-box optimization," *Proc. 7th Int. Conf. on Theory and Application of Cryptology and Inf. Secur., Advances in Cryptology (ASIACRYPT2001)*, pp.239-254, Dec. 2001.

[6] D. Canright. "A very compact S-box for AES," *Proc. 7th Int. Workshop on Cryptographic Hardware and Embedded Systems (CHES2005)*, pp.441-455, Sept. 2005.

[7] P. Hamalainen, T. Alho, M. Hannikainen, T.D. Hamalainen, "Design and Implementation of Low-Area and Low-Power AES Encryption Hardware Core," *Proc. 9th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools (DSD2006)*, pp.577-583, 2006.

[8] Tim Good and Mohammed Benaissa, "Very Small FPGA Application-Specific Instruction Processor for AES," *IEEE Transactions on Circuits and Systems-I: Regular Papers*, vol. 53, no. 7, pp.1477-1486, Jul. 2006.

[9] T. Jarvinen, P. Salmela, P. Hamalainen, J. Takala, "Efficient byte permutation realizations for compact AES implementations," *Proc. 13th European on Signal Processing Conference*, pp.1-4, Sep. 2005.

[10] K. Munusamy, C. Senthilpari, D.C.K. Kho, "A low power hardware implementation of S-Box for Advanced Encryption Standard," *Proc. 11th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pp.1-6, May 2014.

[11] V. Rijmen, "Efficient Implementation of the Rijndael S-Box," Dept. ESAT., Katholieke Universiteit Leuven, Leuven, Belgium, 2006. [Online]. Available: http://www.networkdls.com/Articles/sbox.pdf.

[12] M. T. Sakalli, E. Bulus, A. Sahin, F. Buyuksaraqoglu, "Affine Equivalence in S-boxes," *Proc. 2006 IEEE 14th Signal Processing and Communications Applications,* pp.1-4, Apr. 2006.

[13] D. Canright, "A very compact S-box for AES", In Proc. 7th Int. *Workshop on Cryptographic Hardware and Embedded Systems (CHES2005)*, pp.441-455, Sep., 2005.

[14] D. Canright and L. Batina, "A Very Compact "Perfectly Masked" S-Box for AES," *Proc. ACNS 2008*, vol. 5037, LNCS, pp.446-459, Springer, 2008.

[15] Sanu Mathew et al., "340mV-1.1V, 289Gbps/W, 2090-gate NanoAES Hardware Accelerator with Area-optimized Encrypt/Decrypt $GF(2^4)^2$ Polynomials in 22nm tri-gate CMOS," *2014 Symposium on VLSI Circuits Digest of Technical Papers*, pp.1-2, 2014.

[16] A. Moradi et al., "Pushing the limits: a very compact and a threshold implementation of AES," in *UROCRYPT*, pp.69-88, 2011.

[17] M. Feldhofer, J. Wolkerstorfer and V. Rijmen, "AES implementation on a grain of sand," *IEE Proceedings - Information Security*, vol.152, no.1, pp. 13-20, Oct. 2005.

[18] T. Good and M. Benaissa, "692-nW advanced encryption standard (AES) on a 0.13-µm CMOS," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst*., vol. 18, no. 12, pp. 1753-1757, Dec. 2010.

[19] C. Hocquet et al., "Harvesting the potential of nano-CMOS for lightweight cryptography: an ultra-low-voltage 65 nm AES coprocessor for passive RFID tags," *Springer J. of Crypto. Eng*., vol.1, no.1, pp.79-89, 2011.

[20] Wenfeng Zhao, Yajun Ha, Massimo Alioto, "AES Architectures for Minimum-Energy Operation and Silicon Demonstration in 65nm with Lowest Energy per Encryption," *IEEE International Symposium on Circuits and Systems (ISCAS2015)*, pp.2349-2352, May 2015.