Design of a Specific Instructions Set Processor for AES Algorithm

Karim Shahbazi
Department of Electrical and Computer Engineering
Shahid Beheshti University
Tehran, Iran

Mohammad Eshghi
Department of Electrical and Computer Engineering
Shahid Beheshti University
Tehran, Iran

ABSTRACT
In this paper, a new architecture for Advanced Encryption Standard (AES) Algorithm based on Application Specific Instruction set Processors (ASIP) design technique is proposed. The basic configuration is developed in order to reduce the execution clock pulses for the main specific instructions. According to the improvement of the first register configuration, two ASIPs are designed for AES algorithm. The second ASIP is 89% faster and have 22% less gates than the first proposed design**.**

Keywords
ASIP, AES algorithm, RTL, crypto Processor

1. INTRODUCTION
Today, cryptography has been widely used to achieve information security. Cryptography is a fundamental part of communication and many other digital applications. Data Encryption Standard Algorithm (DES) was the standard encryption algorithm from 1974 till 1999. In 2000, the National Institute of Standards and Technology (NIST) selected the Rijndael algorithm as the new Advanced Encryption Standard (AES) algorithm [1].

AES is a symmetric algorithm that provides an excellent combination of security, performance, efficiency and flexibility [2, 3]. The AES algorithm can also be efficiently implemented on hardware and software platforms [3]. Software implementations of encryption algorithms have the lower cost and higher flexibility in comparison with the hardware implementations. However, the speed of software execution is much lower than the hardware implementation. The AES hardware implementations provide the highest speed for the real-time applications [3]. Many researches have been recently done on different hardware implementations, using ASIC [4, 5] and FPGA technologies [3, 6].

An Application Specific Instruction set Processor (ASIP) is a technique of designing a system on chip with the aim of achieving speed and programmability. The instruction set of an ASIP is selected to operate a specific application. This specialization of the Processor provides a tradeoff among the flexibility, performance and speed of a hardware implementation on FPGA and ASIC platforms. ASIPs have the advantage of both programmability and efficiency at the same time [7].

The implementation and design of Application Specific set Instruction Processor (ASIP) are relatively less explored in the literature [8]. In this work, an ASIP-based design for AES algorithm is proposed. The main register configuration is according to the architecture presented by M. Mano. Then, it is gradually modified to increase speed and flexibility.

The rest of the paper is organized as follows. The AES algorithm is introduced in Section 2, the proposed ASIP designs for AES algorithm is presented in Section 3. The conclusion is in section 4.

2. AN INTRODUCTION TO AES ALGORITHM
Rijndeal is a symmetric encryption and decryption algorithm. Rijndeal crypto a length of 4×32=128, 4×48=192, 4×64= 256 bits input data block. Input is an array of four rows bytes and N_b bytes of columns. Each input data block array called "State". The cipher key size can be 128, 192 or 256 bits. Cipher key puts in an array as input (the number of key columns is called N_k). The Rijndeal for creating the output performs several rounds (the number of rounds is called N_r) according to the input bits and cipher key (table 1). The key expansion is needed to execute the algorithm. Each128-bit of expended key is called Round Key (column number of key expended = $N_b \times (N_r+1)$). AES is the standard algorithm of Rijndeal. The input length in AES is only 128 bits (N_b=4).

Table 1. Number of rounds depending on key and input length [9]

N_r		N_b		
		4	**6**	**8**
N_k	**4**	10	12	14
	6	12	12	14
	8	14	14	14

Figure.1 shows the AES encryption steps. For encryption, there are four basic phases (functions): Add Round Key, Sub Bytes, Shift Rows and Mix Columns. All the four functions are used in any round. However, the first round has only key addition and the last round does not include the Mix Column. In the following, these four functions are described.

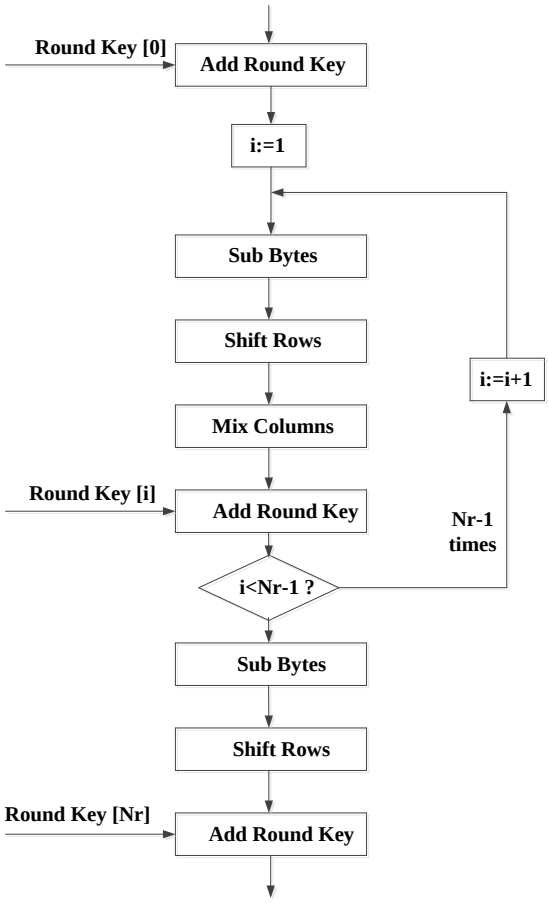

Figure 1. AES encryption block diagram [10]

2.1 Add Round Key function

This function executes a bitwise XOR operation on the state with Round Key.

2.2 Shift Rows function

This is a conversion that operates on the rows of the state. In this function, the bytes of the state are shifted cyclically to the left. The first row does not shift, the second, third, and fourth rows shift cyclically to the left as one, two and three respectively.

2.3 Substitute bytes function

Sub Bytes is the most costly transformation in AES, in both time and area aspects [2]. This function is a nonlinear byte substitution on each byte of the state. In order to execute this function, two processes are used. First, computing the multiplicative inverse and the second, an affine transformation (Figure 2).

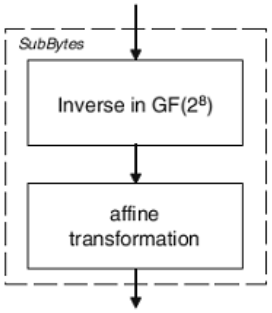

Figure 2: Composition of Sub Bytes [10]

In multiplicative inverse, each byte is computed in $GF(2^8)$ with irreducible polynomial $m(x)=x^8+x^4+x^3+x+1$ (the byte {00} is mapped to itself) and Then the affine transformation is applied. The affine transformation matrix [10, 11] is shown below.

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 10001111 \\ 11000111 \\ 11100011 \\ 11110001 \\ 11111000 \\ 01111100 \\ 00111110 \\ 00011111 \end{bmatrix} \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad (1)$$

Where y_i is the i^{th} bit of a byte. Since there are only 256 representations of one byte, all the byte substitution results can be calculated in advanced. "Sbox" can be implemented by using Galois Field operations and a look up table. Calculating and implementing the multiplicative inverse of elements in $GF(2^8)$ is expensive [4] in aspect of area and time.

Table 2: look up table for the Substitute bytes [1]

x \ y	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	82	c9	7d	fa	59	47	f0	ad	d4	a2	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

2.4 Mix Columns function

Mix Columns is the critical part of the algorithm [11]. The Mix Columns transformation operates on the State, column by column. The multiplication can be written as:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02\ 03\ 01\ 01 \\ 01\ 02\ 03\ 01 \\ 01\ 01\ 02\ 03 \\ 03\ 01\ 01\ 02 \end{bmatrix} \times \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad (2)$$

Or:

$$\begin{aligned} S'_{0,c} &= (\{02\}.\, S_{0,c}) \oplus (\{03\}.S_{1,c}) \oplus S_{2,c} \oplus S_{3,c} \\ S'_{1,c} &= S_{0,c} \oplus (\{02\}.S_{1,c}) \oplus (\{03\}.S_{2,c}) \oplus S_{3,c} \\ S'_{2,c} &= S_{0,c} \oplus S_{1,c} \oplus (\{02\}.S_{2,c}) \oplus (\{03\}.S_{3,c}) \\ S'_{3,c} &= (\{03\}.S_{0,c}) \oplus S_{1,c} \oplus S_{2,c} \oplus (\{02\}.S_{3,c}) \end{aligned} \quad (3)$$

Where $\oplus$ is the XOR operation and '.' is a multiplication modulo the irreducible polynomial $m(x)= x^8+x^4+x^3+x+1$ [11]. More information about AES algorithm is in [1, 10].

3. THE PROPOSED ASIP DESIGN for AES-128 ALGORITHM

Any processor contains of an Arithmetic Logic Unit (ALU), three Buses (Control bus, Address bus and Data bus), some Registers, and a Control Unit (CU). At first the ASIP designer had to design register configuration (Registers and Buses) according to the specific algorithm.

The most fundamental part of an ASIP design, (for controlling the Buses and Registers) is Control Unit. To design a Control Unit, Register Transfer language (RTL) and micro-programming techniques can be used. Each algorithm consists of a number of functions which include series of statements and loops. RTLs are obtained from the functions, according to the hardware architecture (Register Configuration) of the Processor.

Specific instructions of the processor are selected, according to the algorithms in the application. In this work, all AES functions are selected as specific instructions of the proposed processor.

In continue two processor architectures with Specific Instructions for AES-128 algorithm are introduced. At first, the Mano architecture [12] is selected as the basic and initial Register Configuration for the proposed ASIP, and its control unit is designed. Then, in order to increase the speed of processor, a new Register Configuration, including some specific look-up-tables, is proposed. Finally, an offline key-expansion memory is designed and added to the architectures, in order to increase the speed of execution of instructions.

In section 3.1 the first proposed architecture for AES ASIP, and in 3.2 its second architecture is presented. Section 3.3 contains a comparison between the performance of these two architecture based on their area and time criteria.

3.1 First proposed Register Configuration for AES ASIP

In this Register Configuration, Mano architecture, with basic ALU functions (AND, OR, SHIFT RIGHT, SHIFT LEFT, ADD, XOR) is used. The only different is the data bus and registers. Because of using multiplication of elements of $GF(2^8)$ (modulo $x^8+x^4+x^3+x+1$) and the number of Up Codes, the Data Bus and Registers are considered to be 17-bits.

A Look Up Table (LUT) is also added to the Mano architecture. According to the explanations given about Sub Bytes, Each input byte as the address of register bank should to substitute with the value from a register bank (LUT). One 8 to 256 decoder is used for controlling this register bank. The decoder needs an Enable signal, to be activated when executing the Sub Bytes instruction.

3.2 Second proposed Register Configuration for AES ASIP

In the second proposed Register Configuration, to speed up the execution of Mix Column instruction, the ALU is improved in such a way that it can execute multiplied modulo $x^8+x^4+x^3+x+1$ in one clock cycle. In this architecture for more access to data and addresses, the temporary registers have been increased. Also ALU can access to data bus through an accumulator register AC0 (Figure 4).

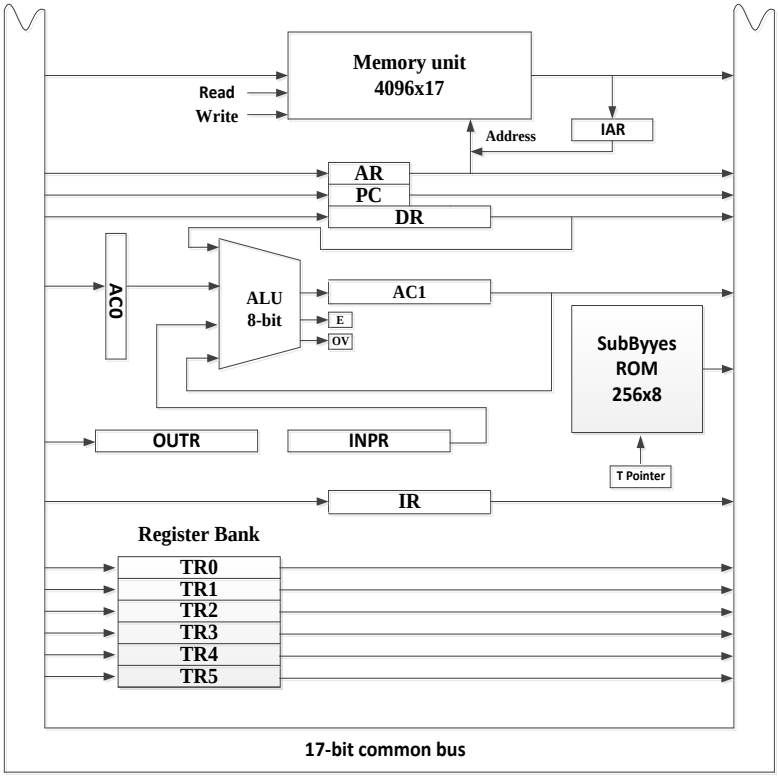

Figure 4: The Second proposed register configuration for AES ASIP

In order to design the control unit of the proposed processor, we used RTL lever description. For example, some of needed RTLs for execution of the "Add-Round-Key" instruction are shown below.

T_{i+2}.MC_1.Add-Round-Key: DR←MEM (AR);
T_{i+2}.MC_2.Add-Round-Key: AR←TR1;
T_{i+2}.MC_3.Add-Round-Key: AC0←MEM (AR);
T_{i+2}.MC_4.Add-Round-Key: AC1←AC0 Xor DR;
T_{i+2}.MC_5.Add-Round-Key: AR←TR0;
T_{i+2}.MC_6: MEM (AR)←AC1, AR,TR0,TR1++, MC←0

3.3 Performance evaluation of two proposed AES ASIP designs

Each RTL is executed in one clock. For a rough estimation, it is assumed that each RTL needs at least two gates in Control Unit and every ROM cell has three gates. Thus, the number of gates used for first configuration is about 1527 and for second configuration is about 1188 gates. The comparison results between the number of required clock pulses to execute each specific instruction and the approximately number of gates in the different versions of AES Processor are shown in table 3.

Considering this table, it is clear that the AES with 2nd Register Configuration uses fewer gates and executes in much less clock pulses. The entity and input/output buses of AES processor is shown in Figure 5.

The general instruction Set of the proposed AES processor with their symbolic descriptions is listed in the Table 4.

TABLE 3. COMPARISON NUMBER OF CLOCK PULSE AND GATES OF SPECIFIC INSTRUCTIONS FOR DIFFERENT REGISTER CONFIGURATION

Specific Instructions	# Clock Pulse	
	1st RC	2nd RC
Sub Bytes	48	48
Shift Rows	110	110
Mix columns	9410	712
Add Round Key	114	114
Total clock pulses for one execute	87524	9242
	# Number Of Gates	
Total approximate usage Gates	1527	1188

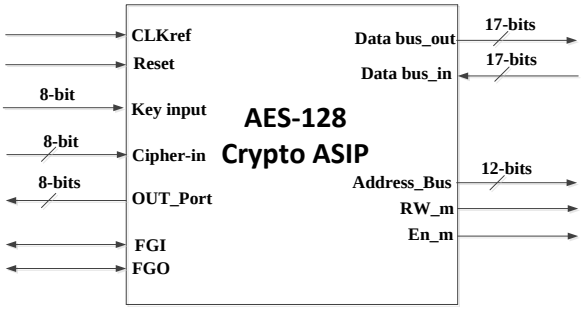

Figure 5: I/O signals and data buses for AES Crypto processor

Table 4. General instructions for the proposed ASIP

General Instructions	Description
Mult	AC←AC Mult DR (multiplied modulo $x^8+x^4+x^3+x+1$)
ADD	AC←AC + M[AR], E←Cout
SUB	AC←AC - M[AR]
ISZ	M[AR]←M[AR] + 1, If M[AR]+1=0 then PC←PC+1
LDA	AC←M[AR]
STA	M[AR]←AC
LIA	AC←M[M[AR]]
SIA	M[M[AR]]←AC
BUN	PC←IR(0-11)
JPA	If (AC(31)=0) then (PC←IR(0-11))
JZA	If (AC=0) then (PC←IR(0-11))
JZE	If (E=0) then (PC←IR(0-11))
JPV	If (OV=1) then (PC←IR(0-11))
OR	AC←AC∨ M[AR]
AND	AC←AC∧ M[AR]
XOR	AC←AC⊕ M[AR]
OUTPUT	OUTR←AC, FGO←0
INPUT	AC←INPR, FGI←0
CMA	AC←¬AC
ADDAD	AC←AC + DR
SHR	AC←SHR(AC)
SHL	AC←SHL(AC)
ROR	AC←ROR(AC)
ROL	AC←ROL(AC)
CLRA	AC←0
CLRV	OV←0
CLRE	E←0
CME	E←¬E
RET	PC←RETURN ADDRESS
CALL	PC←IR(5-16), RETURN ADDRESS←PC
MOVAD	DR←AC
MOVDA	AC←DR
ORAD	AC←AC∨ DR
INCA	AC←AC + 1
DECA	AC←AC – 1
NOP	Nothing
ADDOP	AC←OP1 + OP2
SKO	If (FGO=1) then (PC←PC+1)
SKI	If (FGI=1) then (PC←PC+1)
HALT	S←0 (S is a start-stop flip-flop)

4. CONCLUSION

In this paper an AES processor with ASIP technique is designed. The instruction set consists of general instructions and specific cryptographic ones. The initial Register Configuration is considered, according to Mano architecture. The Mano Configuration has been improved by adding extra registers and a Look up Table. Also Mano ALU is improved to execute multiplied modulo $x^8+x^4+x^3+x+1$ in one clock.

According to table 3, the second proposed architecture, due to improvement of its Register Configuration and Control Unit, benefits from about 22% few gates hardware and 89% faster time than the first Register Configuration.

5. REFERENCES

[1] P. FIPS, "197," Advanced Encryption Standard (AES), vol. 26, 2001.

[2] F. Hossain, M. Ali, M. Al Abedin Syed, "A very low power and high throughput AES processor," in Computer and Information Technology (ICCIT), 2011 14th International Conference on, 2011, pp. 339-343.

[3] M.-H. Jing, Z.-H. Chen, J.-H. Chen, Y.-H. Chen, , "Reconfigurable system for high-speed and diversified AES using FPGA," Microprocessors and Microsystems, vol. 31, pp. 94-102, 2007.

[4] A. Hodjat and I. Verbauwhede, "Minimum area cost for a 30 to 70 Gbits/s AES processor," in VLSI, 2004. Proceedings. IEEE Computer society Annual Symposium on, 2004, pp. 83-88.

[5] H. Yin, H. Debiao, K. Yong, F. Xiande, "High-speed ASIC implementation of AES supporting 128/192/256 bits," in Test and Measurement, 2009. ICTM'09. International Conference on, 2009, pp. 95-98.

[6] J. M. Granado-Criado, M.A.Vega-Rodríguez, J.M. Sánchez-Pérez, J.A. Gómez-Pulido, "A new methodology to implement the AES algorithm using partial and dynamic reconfiguration," INTEGRATION, the VLSI journal, vol. 43, pp. 72-80, 2010.

[7] R. F. Mirzaee, M.Eshghi, K. Navi, "Design and Implementation of an ASIP-Based Crypto Processor for IDEA and SAFER K-64," International Journal of Design, Analysis and Tools for Integrated Circuits and Systems (IJDATICS), vol. 3, p. 21, 2012.

[8] R. Chen, Z. Jia, Y. Li, H. Xia, X. Li, , "The application specific instruction processor for AES," in Electronics Computer Technology (ICECT), 2011 3rd International Conference on, 2011, pp. 394-396.

[9] B. Gladman, "A specification for Rijndael, the AES algorithm," at fp. gladman. plus. com/cryptography_technology/rijndael/aes. spec, vol. 3.1, pp. 1-29, 2001.

[10] K. Gaj, P. Chodowiec, , "FPGA and ASIC implementations of AES," in Cryptographic Engineering, ed: Springer, 2009, pp. 235-294.

[11] J. M. Granado, M.A.Vega-Rodríguez, J.M. Sánchez-Pérez, J.A. Gómez-Pulido, , "IDEA and AES, two cryptographic algorithms implemented using partial and dynamic reconfiguration," Microelectronics Journal, vol. 40, pp. 1032-1040, 2009.

[12] M. M. Mano, Computer system architecture: Prentice-Hall Englewood Cliffs, NJ, 1993.