Hardware Implementation of Mix Column Step in AES

Pratap Kumar Dakua
Department of ECE Engineering
CUTM, Paralakhemundi Campus, Odisha, India

Manoranjan Pradhan
Department of ETC Engineering
VSS University of Technology, Odisha, India

Subba Rao Polamuri
Department of ECE Engineering
CUTM, Paralakhemundi Campus, Odisha, India

ABSTRACT

This document gives the hardware implementation of Mix Column step in AES encryption process. The AES encryption process consists of several transformation steps such as byte substitution, shift rows, mix column and addition of round key operation step. There are two aspects to perform mix column step in AES is presented. The total operation is coded with VERILOG, synthesized and simulated using Xilinx ISE 10.1.

Keywords

AES; VLSI; Data Security; Cryptography.

1. INTRODUCTION

To make a data in hidden form, it is necessary to change the data from its original form. Cryptography is the art of representation of data from its original form to another form which is not readable. For this purpose several algorithms are used in cryptography. AES is a cryptographic algorithm used to protect electronic data. AES is a symmetric block cipher which is capable of using cryptographic keys of 128, 192 and 256 bits to encrypt data block of 128 bits [1-2]. In Symmetric key cryptography a shared key used for both encryption and decryption process. The AES encryption process of AES-128 bit consists of 10 rounds. Each round performs different operation such as shift rows, byte substitution, mix column step and addition of round key operations. The details of mix column step are presented in next section.

2. MIXCOLUMN OPERATION

For mix column operation each column is mixed independent of the other. It needs matrix multiplication. The output of this step is matrix multiplication of the old values and a constant matrix. There are two aspects of this step. first explains which parts of the state are multiplied against which parts of the matrix. The second explains how this multiplication is implemented in Galois Field [3].

2.1 MATRIX MULTIPLICATION

The multiplication is performed one column at a time (i.e. on 4 bytes at a time). Each value in the column is multiplied against every value of the matrix (i.e. 16 total multiplications are performed). The results of these multiplications are XORed together to produce only 4 resulting bytes for the next step. That means, we together have 4 bytes input, 16 multiplications, 12 XORs and 4 bytes output. The multiplication is performed one matrix row at a time against each value of a state column. For example, consider the matrix as shown in Fig. 1.

$$\begin{bmatrix} 11 & FB & B9 & CF \\ ED & FE & F7 & AD \\ 7C & 97 & 89 & BC \\ A4 & BE & CD & A0 \end{bmatrix}$$

Fig.1 Matrix Multiplication

And the state as:

$$\begin{bmatrix} BE & 96 & A1 & 35 \\ 53 & 5C & E4 & D8 \\ DD & 7C & 84 & 8B \\ B4 & A7 & F2 & 17 \end{bmatrix}$$

Fig.2 The State

And the 16 Byte state arrays are:

$$\begin{bmatrix} B1 & B5 & B9 & B13 \\ B2 & B6 & B10 & B14 \\ B3 & B7 & B11 & B15 \\ B4 & B8 & B12 & B16 \end{bmatrix}$$

Fig.3. 16-Byte state array

Therefore the total expression can be found as:

$$\begin{bmatrix} B1 & B5 & B9 & B13 \\ B2 & B6 & B10 & B14 \\ B3 & B7 & B11 & B15 \\ B4 & B8 & B12 & B16 \end{bmatrix} =$$

$$\begin{bmatrix} 11 & FB & B9 & CF \\ ED & FE & F7 & AD \\ 7C & 97 & 89 & BC \\ A4 & BE & CD & A0 \end{bmatrix} \times \begin{bmatrix} BE & 96 & A1 & 35 \\ 53 & 5C & E4 & D8 \\ DD & 7C & 84 & 8B \\ B4 & A7 & F2 & 17 \end{bmatrix}$$

The first result byte i.e. B1 is calculated by multiplying four values of the first row of the matrix. The result of each multiplication is then XORed to produce one byte. For this, the following calculation is used:

B1= (11 * BE) XOR (FB * 53) XOR (B9 * DD) XOR (CF * B4)

Next, the second result byte is calculated by multiplying the same four values of the state column against four values of the second row of the matrix. The result of each multiplication is then XORed to produce one byte.

B2= (ED * BE) XOR (FE * 53) XOR (F7 * DD) XOR (AD * B4)

This procedure is repeated again with the next column of the state, until there are no more state columns.

To summarize:

The first column will include state bytes 1-4 and will be multiplied against the matrix in the following manner:

B1= (11 * BE) XOR (FB * 53) XOR (B9 * DD) XOR (CF * B4)

B2= (ED * BE) XOR (FE * 53) XOR (F7 * DD) XOR (AD * B4)

B3= (7C * BE) XOR (97 * 53) XOR (89 * DD) XOR (BC * B4)

B4= (A4 * BE) XOR (BE * 53) XOR (CD * DD) XOR (A0 * B4)

(B1= specifies the first byte of the state)

The second column will be multiplied against the second row of the matrix in the following manner.

B5= (11 * 96) XOR (FB * 5C) XOR (B9 * 7C) XOR (CF * A7)

B6= (ED * 96) XOR (FE * 5C) XOR (F7 * 7C) XOR (AD * A7)

B7= (7C * 96) XOR (97 * 5C) XOR (89 * 7C) XOR (BC * A7)

B8= (A4 * 96) XOR (BE * 5C) XOR (CD * 7C) XOR (A0 * A7)

And so on until all columns of the state are exhausted.

2.2 GALOIS FIELD MULTIPLICATION

The multiplication mentioned above is performed over a Galois field. The implementation of this multiplication can be done quite easily with the use of the following two tables, shown in hexadecimal formats. look up of the L table, followed by the addition of the results, followed by a look up of the E table.

All numbers being multiplied using the Mix Column function converted to Hex will form a maximum of 2 digit Hex number. Here in the table we use the first digit on the vertical index and the second number on the horizontal index. If the value being multiplied is composed of only one digit we use 0 on the vertical index The above example the two Hex values being multiplied 11 * BE, we first lookup L (11) index which returns 04 and then lookup L (BE) which returns 29. Once the L table lookup is complete we can then simply add the numbers together. The trick is that if the addition result is greater than FF, we subtract FF from the addition result.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	01	03	05	0F	11	33	55	FF	1A	2E	72	96	A1	F8	13	35
1	5F	E1	38	48	D8	73	95	A4	F7	02	06	0A	1E	22	66	AA
2	E5	34	5C	E4	37	59	EB	26	6A	BE	D9	70	90	AB	E6	31
3	53	F5	04	0C	14	3C	44	CC	4F	D1	68	B8	D3	6E	B2	CD
4	4C	D4	67	A9	E0	3B	4D	D7	62	A6	F1	08	18	28	78	88
5	83	9E	B9	D0	6B	BD	DC	7F	81	98	B3	CE	49	DB	76	9A
6	B5	C4	57	F9	10	30	50	F0	0B	1D	27	69	BB	D6	61	A3
7	FE	19	2B	7D	87	92	AD	EC	2F	71	93	AE	E9	20	60	A0
8	FB	16	3A	4E	D2	6D	B7	C2	5D	E7	32	56	FA	15	3F	41
9	C3	5E	E2	3D	47	C9	40	C0	5B	ED	2C	74	9C	BF	DA	75
A	9F	BA	D5	64	AC	EF	2A	7E	82	9D	BC	DF	7A	8E	89	80
B	9B	B6	C1	58	E8	23	65	AF	EA	25	6F	B1	C8	43	C5	54
C	FC	1F	21	63	A5	F4	07	09	1B	2D	77	99	B0	CB	46	CA
D	45	CF	4A	DE	79	8B	86	91	A8	E3	3E	42	C6	51	F3	0E
E	12	36	5A	EE	29	7B	8D	8C	8F	8A	86	94	A7	F2	0D	17
F	39	4B	DD	7C	84	97	A2	FD	1C	24	6C	B4	C7	52	F6	01

Fig. 4 E Table

The result of the multiplication is actually the output of a

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		00	19	01	32	02	1A	C6	4B	C7	1B	68	33	EE	DF	03
1	64	04	E0	0E	34	8D	81	EF	4C	71	08	C8	F8	69	1C	C1
2	7D	C2	1D	B5	F9	B9	27	6A	4D	E4	A6	72	9A	C9	09	78
3	65	2F	8A	05	21	0F	E1	24	12	F0	82	45	35	93	DA	8E
4	96	8F	D8	BD	36	D0	CE	94	13	5C	D2	F1	40	46	83	38
5	66	DD	FD	30	BF	06	8B	62	B3	25	E2	98	22	88	91	10
6	7E	6E	48	C3	A3	B6	1E	42	3A	6B	28	54	FA	85	3D	BA
7	2B	79	0A	15	9B	9F	5E	CA	4E	D4	AC	E5	F3	73	A7	57
8	AF	58	A8	50	F4	EA	D6	74	4F	AE	E9	D5	E7	E6	AD	E8
9	2C	D7	75	7A	EB	16	0B	F5	59	CB	5F	B0	9C	A9	51	A0
A	7F	0C	F6	6F	17	C4	49	EC	D8	43	1F	2D	A4	76	7B	B7
B	CC	BB	3E	5A	FB	60	B1	86	3B	52	A1	6C	AA	55	29	9D
C	97	B2	87	90	61	BE	DC	FC	BC	95	CF	CD	37	3F	5B	D1
D	53	39	84	3C	41	A2	6D	47	14	2A	9E	5D	56	F2	D3	AB
E	44	11	92	D9	23	20	2E	89	B4	7C	B8	26	77	99	E3	A5
F	67	4A	ED	DE	C5	31	FE	18	0D	63	8C	80	C0	F7	70	07

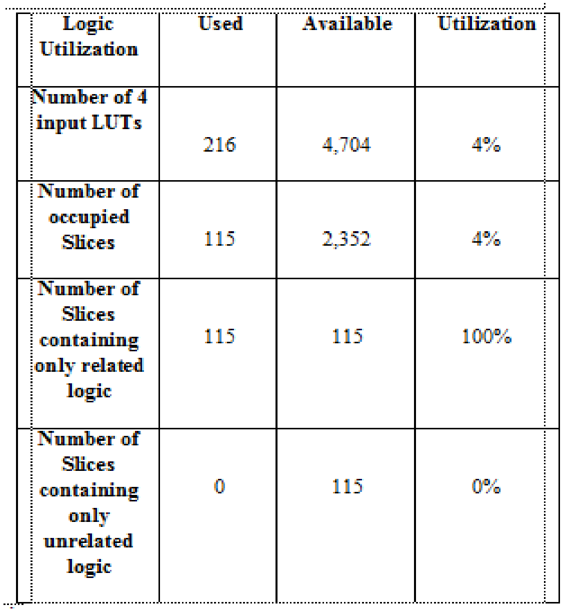
Fig. 5 L Table

For example B7+4B= 102. Because 102>FF, we perform: 102-FF which gives us 03. The last step is to lookup the addition result on the E table. Again we take the first digit to lookup the vertical index and the second digit to lookup the horizontal index. For example E (A4) = AC. Therefore, the result of multiplying 11 * BE over a Galois Field is AC.

3. EXPERIMENTAL RESULT

The Verilog code of Mix Column operation of AES process is synthesized and simulated using Xilinx ISE 10.1.

Table 1. Table for Logic Utilization



Logic Utilization	Used	Available	Utilization
Number of 4 input LUTs	216	4,704	4%
Number of occupied Slices	115	2,352	4%
Number of Slices containing only related logic	115	115	100%
Number of Slices containing only unrelated logic	0	115	0%

It is implemented on xc2s200-6pq208.The data input to mix column block is 6353e08c0960e104cd70b751bacad0e7, (as provided by FIPS of NIST) which gives 32-bit output as 5f72641557f5bc92f7be3b291db9f91a.[4] The output value is correct as provided by FIPS (Federal information processing standard).Figure 6 Shows the RTL schematic of Mix Column Operation. It has two input ports and one output port.

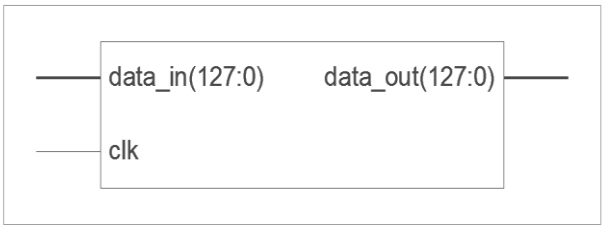


Fig. 6 RTL view of Mix Column operation

Figure 7 displays the simulation waveform of Mix Column operation.

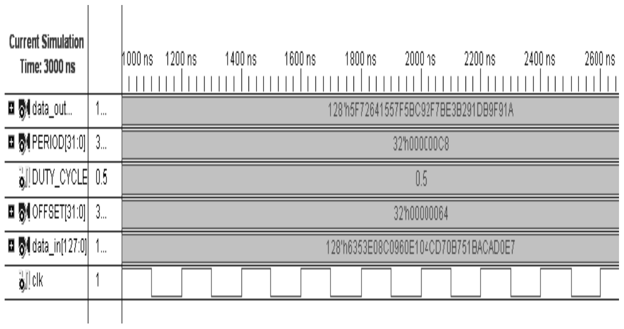


Fig. 7 Simulation waveform of Mix Column Step

4. CONCLUSION

The mix column step of AES process is coded with Verilog and synthesized using Xilinx ISE 10.1. The mix column process is implemented using xc2s200-6pq208 FPGA Xilinx device. Each step of mix column operation is tested with some of the sample vectors provided by NIST and the output results are correct.

5. REFERENCES

[1] J.Daemen and V.Rijmen, "AES Proposal:Rijndael," AES Algorithm Submission,September3,1999.

[2] FIPS 197, "Advanced Encryption Standard (AES) ", November 26, 2001.

[3] A. Kahate, Cryptography and Network Security, 2nd Edition, Tata Mc Graw Hill, 2008.

[4] NIST: Specification for the Advanced Encryption Standard (AES). Technical Report FIPS PUB 197, National Institute of Standards and Technology (NIST) (2001.

[5] B. Schneier, Applied Cryptography Second Edition, John Wiley & Sons, 1996

[6] W. Stallings, Cryptogrphy and Network Security: Principles and Practices, 3rd edition, 2003

[7] Tilborg, Henk C.A.van., Fundamentals of Cryptography: A Professional Reference and Interactive Tutorial, New York. Kluwer Academic publishers, 2002.