SubBytes Transform circuit for AES Cipher (Version 1.0)

Tom Wada, Prof of the University of the Ryukyus, Information Engineering Dept.

---

[0] Introduction

This year's design target is the SybBytes transform circuit, which is one of the key data transformation used in ADVANCED ENCRYPTION STANDARD (AES).

AES encrypts information by repeatedly using the following four kinds of data transformation, ShiftRows, SubBytes, MixColumns, AddRoundKey. The standard document is available in the following URL.

http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

Since this contest is mainly for University students, the code size in the task is relatively small to let student to design easily. Then the design target is not entire AES encryption /decryption system. The design target is just SubBytes transform circuit. The required function is simple such as the transformation from 8bit data to 8bit data. Then the transformation can be implemented using 256 word x 8 bit Read Only Memory (ROM). However, since the SybBytes transform algorithm is based on Galois Field Inverse operation and simple affine transformation, the function can be realized by small low-power digital circuit once good algorithm is applied for the design.

The requirements of the design is to write HDL (VHDL or Verilog HDL) and to synthesize digital circuits using Synopsys design analyzer or any other EDA tools. Making FPGA is also optional but our judges love to see your FPGA designs.

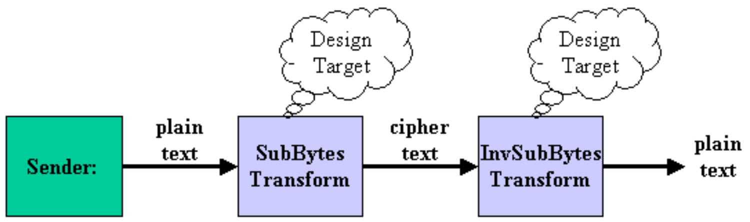


Figure1 System Diagram

The figure 1 shows the system diagram. The system is divided into three blocks such as Sender, SubBytes Transform, and Inverse SubBytes Transform. Sender generates 1 byte (8 bit) data serially. SubBytes Transform encipher the 8bit. Inverse SubBytes Transform decipher the encrypted 8bit to plain byte. The design target is the transform circuit which can both execute SubBytes and Inverse SubBytes transformations by mode switch signal.

---

[1] SubBytes Transform

SubBytes transform is a simple transform which converts 8bit data to other 8 bit data. For example, 8 bit data "00000000" is transformed into "01100011". What is important is that different byte data are always transformed into different 8 bit data. If this condition is violated, the transformed data can not be recovered. Such system can not be used for data encryption. This required condition does fit Galois Field characteristics.

The definition of SubBytes Transformation is the serial transformation of the following two transformation.

1. Take the multiplicative inverse in Galois Field GF(2^8), the element 8bit "00000000" = hex {00} is mapped to itself.

   Inverse Table is shown as follows. For example, 8bit number "01010011" is {53} in hexadecimal. Then consult table by X=5, Y=3, then the output {CA} in hex is obtained. {CA} is "11001010" in binary format.

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

E	B1	0D	D6	EB	C6	0E	CF	AD	08	4E	D7	E3	5D	50	1E	B3
F	5B	23	38	34	68	46	03	8C	DD	9C	7D	A0	CD	1A	41	1C

Table 1. multiplicative inverse table in GF(2^8)

Irreducible polynomial is as follows.

$$m(x) = x^8 + x^4 + x^3 + x + 1$$

2. Then apply the affine transformation over GF(2). Binary "00000000" = Hex {00} will be transformed into "01100011"={63}.

$$\begin{bmatrix} b7\\ b6\\ b5\\ b4\\ b3\\ b2\\ b1\\ b0 \end{bmatrix} = \begin{bmatrix} 1&1&1&1&1&0&0&0\\ 0&1&1&1&1&1&0&0\\ 0&0&1&1&1&1&1&0\\ 0&0&0&1&1&1&1&1\\ 1&0&0&0&1&1&1&1\\ 1&1&0&0&0&1&1&1\\ 1&1&1&0&0&0&1&1\\ 1&1&1&1&0&0&0&1 \end{bmatrix} \times \begin{bmatrix} a7\\ a6\\ a5\\ a4\\ a3\\ a2\\ a1\\ a0 \end{bmatrix} \oplus \begin{bmatrix} 0\\ 1\\ 1\\ 0\\ 0\\ 0\\ 1\\ 1 \end{bmatrix}$$

If $(a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)$={CA}, then the computation over GF(2) is performed as follows.

$$\begin{bmatrix} b7\\ b6\\ b5\\ b4\\ b3\\ b2\\ b1\\ b0 \end{bmatrix} = \begin{bmatrix} 1&1&1&1&1&0&0&0\\ 0&1&1&1&1&1&0&0\\ 0&0&1&1&1&1&1&0\\ 0&0&0&1&1&1&1&1\\ 1&0&0&0&1&1&1&1\\ 1&1&0&0&0&1&1&1\\ 1&1&1&0&0&0&1&1\\ 1&1&1&1&0&0&0&1 \end{bmatrix} \times \begin{bmatrix} 1\\ 1\\ 0\\ 0\\ 1\\ 0\\ 1\\ 0 \end{bmatrix} \oplus \begin{bmatrix} 0\\ 1\\ 1\\ 0\\ 0\\ 0\\ 1\\ 1 \end{bmatrix} = \begin{bmatrix} 1\cdot1\oplus1\cdot1\oplus1\cdot0\oplus1\cdot0\oplus1\cdot1\oplus0\cdot0\oplus0\cdot1\oplus0\cdot0\oplus0\\ \cdots\\ \cdots\\ \cdots\\ \cdots\\ \cdots\\ \cdots\\ \cdots \end{bmatrix} = \begin{bmatrix} 1\\ 1\\ 1\\ 0\\ 1\\ 1\\ 0\\ 1 \end{bmatrix}$$

{CA} will be transformed into "11101101 ={ED}.

Totally, {53} is converted to {CA} by Inversion, and then transformed into {ED} by affine transform. These serial transform is SubBytes transformation.

[2] Inverse SubBytes Transform

Inverse SubBytes Transformation is the reverse operation of SubBytes transformation. Then the following two serial transformation will be executed.

1. The inverse affine transformation is executed. The inverse transformation will be given in the following equation.

$$\begin{bmatrix} a7\\ a6\\ a5\\ a4\\ a3\\ a2\\ a1\\ a0 \end{bmatrix} = \begin{bmatrix} 0&1&0&1&0&0&1&0\\ 0&0&1&0&1&0&0&1\\ 1&0&0&1&0&1&0&0\\ 0&1&0&0&1&0&1&0\\ 0&0&1&0&0&1&0&1\\ 1&0&0&1&0&0&1&0\\ 0&1&0&0&1&0&0&1\\ 1&0&1&0&0&1&0&0 \end{bmatrix} \times \begin{bmatrix} b7\\ b6\\ b5\\ b4\\ b3\\ b2\\ b1\\ b0 \end{bmatrix} \oplus \begin{bmatrix} 0\\ 0\\ 0\\ 0\\ 0\\ 1\\ 0\\ 1 \end{bmatrix}$$

2. Then perform the same multiplicative inverse in Galois Field GF(2^8) according to the table 1.

Since both SubBytes and Inverse SubBytes transformations needs the multiplicative inverse in Galois Field GF(2^8), the design target function can be realized in the architecture shown in figure 2.

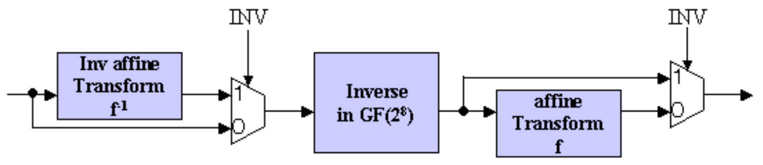


Figure 2. Architecture Example

When INV signal is '0', input data first goes to Inverse Block then transferred to affine transform, and when INV signal is asserted (='1'), input data is first transformed in Inv Affin Block and later Inverse in GF(2^8) is caluculated.

Since the affine transform is just a calculation using ANDs and EXORs, how calculate the inverse in GF(2^8) should be weill considered in the design task.

[3] How Inverse in GF(2^8) is calculated

Most easy implementation of Inverse calculation in GF(2^8) is to use 256 word x 8 bit ROM which possesses the data in Table 1.

However, the ROM implementation is not intersting in the design contest.

Another implementation idea will be explained as follows.

Suppose input is y in GF(2^8) , y^{-1} has to be calculated. In G(2^8) ,

$$y^{2^8-1} = y^{255} = 1$$

is satisfied. Then,

$$y^{-1} = y^{-1} \cdot y^{255} = y^{254}$$

Consequently, y^{254} is equivalent to y^{-1}. Figure 3 shows an Inversion circuit based on Itoh and Tsujii's algorithm.

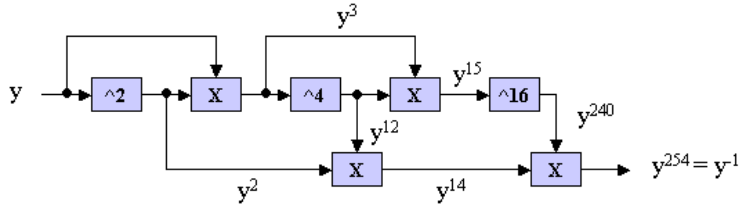


Figure 3. An Inversion circuit based on Itoh and Tsujii's algorithm

---

[4] One example of multiply circuit in GF(2^8)

According to the preceding section, inverse can be computed by the repetitive use of multiply circuits.

Suppose the multiply of two 8-bit value A= (a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0), B=(b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0). Since those 8 bit value can be expressed in polynomials in Galois Field such as

$$A(x) = a_7 \cdot x^7 + a_6 \cdot x^6 + a_5 \cdot x^5 + a_4 \cdot x^4 + a_3 \cdot x^3 + a_2 \cdot x^2 + a_1 \cdot x^1 + a_0$$
$$B(x) = b_7 \cdot x^7 + b_6 \cdot x^6 + b_5 \cdot x^5 + b_4 \cdot x^4 + b_3 \cdot x^3 + b_2 \cdot x^2 + b_1 \cdot x^1 + b_0$$

Then C(x) = A(x) * B(x) gives

$$C(x) = c_{14} \cdot x^{14} + c_{13} \cdot x^{13} + c_{12} \cdot x^{12} + c_{11} \cdot x^{11} + c_{10} \cdot x^{10} + c_9 \cdot x^9 + c_8 \cdot x^8 +$$
$$c_7 \cdot x^7 + c_6 \cdot x^6 + c_5 \cdot x^5 + c_4 \cdot x^4 + c_3 \cdot x^3 + c_2 \cdot x^2 + c_1 \cdot x^1 + c_0$$

$$c_{14} = a_7 \cdot b_7$$
$$c_{13} = a_7 \cdot b_6 \oplus a_6 \cdot b_7$$
$$c_{12} = a_7 \cdot b_5 \oplus a_6 \cdot b_6 \oplus a_5 \cdot b_7$$
$$c_{11} = a_7 \cdot b_4 \oplus a_6 \cdot b_5 \oplus a_5 \cdot b_6 \oplus a_4 \cdot b_7$$
$$c_{10} = a_7 \cdot b_3 \oplus a_6 \cdot b_4 \oplus a_5 \cdot b_5 \oplus a_4 \cdot b_6 \oplus a_3 \cdot b_7$$
$$c_9 = a_7 \cdot b_2 \oplus a_6 \cdot b_3 \oplus a_5 \cdot b_4 \oplus a_4 \cdot b_5 \oplus a_3 \cdot b_6 \oplus a_2 \cdot b_7$$
$$c_8 = a_7 \cdot b_1 \oplus a_6 \cdot b_2 \oplus a_5 \cdot b_3 \oplus a_4 \cdot b_4 \oplus a_3 \cdot b_5 \oplus a_2 \cdot b_6 \oplus a_1 \cdot b_7$$
$$c_7 = a_7 \cdot b_0 \oplus a_6 \cdot b_1 \oplus a_5 \cdot b_2 \oplus a_4 \cdot b_3 \oplus a_3 \cdot b_4 \oplus a_2 \cdot b_5 \oplus a_1 \cdot b_6 \oplus a_0 \cdot b_7$$
$$c_6 = a_6 \cdot b_0 \oplus a_5 \cdot b_1 \oplus a_4 \cdot b_2 \oplus a_3 \cdot b_3 \oplus a_2 \cdot b_4 \oplus a_1 \cdot b_5 \oplus a_0 \cdot b_6$$
$$c_5 = a_5 \cdot b_0 \oplus a_4 \cdot b_1 \oplus a_3 \cdot b_2 \oplus a_2 \cdot b_3 \oplus a_1 \cdot b_4 \oplus a_0 \cdot b_5$$
$$c_4 = a_4 \cdot b_0 \oplus a_3 \cdot b_1 \oplus a_2 \cdot b_2 \oplus a_1 \cdot b_3 \oplus a_0 \cdot b_4$$
$$c_3 = a_3 \cdot b_0 \oplus a_2 \cdot b_1 \oplus a_1 \cdot b_2 \oplus a_0 \cdot b_3$$
$$c_2 = a_2 \cdot b_0 \oplus a_1 \cdot b_1 \oplus a_0 \cdot b_2$$
$$c_1 = a_1 \cdot b_0 \oplus a_0 \cdot b_1$$
$$c_0 = a_0 \cdot b_0$$

The result is 14th order polynomials. Be sure to remember that all coefficients follows GF(2) rule. Then multiply is replaced withs AND logic, addition/subtraction are replaced with exclusive-or logics.

Assuming Irreducible polynomial = 0, the following relations are obtained.

$$m(x) = x^8 + x^4 + x^3 + x + 1 = 0$$
$$x^8 = x^4 + x^3 + x + 1$$
$$x^9 = x^5 + x^4 + x^2 + x$$
$$x^{10} = x^6 + x^5 + x^3 + x^2$$
$$x^{11} = x^7 + x^6 + x^4 + x^3$$
$$x^{12} = x^8 + x^7 + x^5 + x^4 = (x^4 + x^3 + x + 1) + x^7 + x^5 + x^4 = x^7 + x^5 + x^3 + x + 1$$
$$x^{13} = x^8 + x^6 + x^4 + x^2 + x = (x^4 + x^3 + x + 1) + x^6 + x^4 + x^2 + x = x^6 + x^3 + x^2 + 1$$
$$x^{14} = x^7 + x^4 + x^3 + x$$

By applying those relations to C(x), C(x) can be modified into 7th order polynomial D(x).

$$D(x) = d_7 \cdot x^7 + d_6 \cdot x^6 + d_5 \cdot x^5 + d_4 \cdot x^4 + d_3 \cdot x^3 + d_2 \cdot x^2 + d_1 \cdot x^1 + d_0$$

$$\begin{aligned}
d_7 &= c_7 \oplus c_{11} \oplus c_{12} \oplus c_{14} \\
d_6 &= c_6 \oplus c_{10} \oplus c_{11} \oplus c_{13} \\
d_5 &= c_5 \oplus c_9 \oplus c_{10} \oplus c_{12} \\
d_4 &= c_4 \oplus c_8 \oplus c_9 \oplus c_{11} \oplus c_{14} \\
d_3 &= c_3 \oplus c_8 \oplus c_{10} \oplus c_{11} \oplus c_{12} \oplus c_{13} \oplus c_{14} \\
d_2 &= c_2 \oplus c_9 \oplus c_{10} \oplus c_{13} \\
d_1 &= c_1 \oplus c_8 \oplus c_9 \oplus c_{12} \oplus c_{14} \\
d_0 &= c_0 \oplus c_8 \oplus c_{12} \oplus c_{13}
\end{aligned}$$

Since D(x) expresses GF(2^8) , the multiplication result of two numbers in GF(2^8) A= (a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)、 B=(b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0) will be D=(d_7, d_6, d_5, d_4, d_3, d_2, d_1, d_0) .

In another words, multiplication in GF(2^8) can be implemented using AND and EXOR logics.

(NOTE) The above equation is not fully debugged. If you use the equation, please check by yourself.

---

[5] LEVEL 1 Task for beginners

The beginners' task has the following pin lists.

SubBytes			
signal name	in or out	bit width	explanation
CLK	IN	1	clock input
RESET	IN	1	assertion '1' means reset
XIN	IN	8	input data
INV	IN	1	'0' means SubBytes, '1' means Inverse SubBytes
YOUT	OUT	8	transformed output

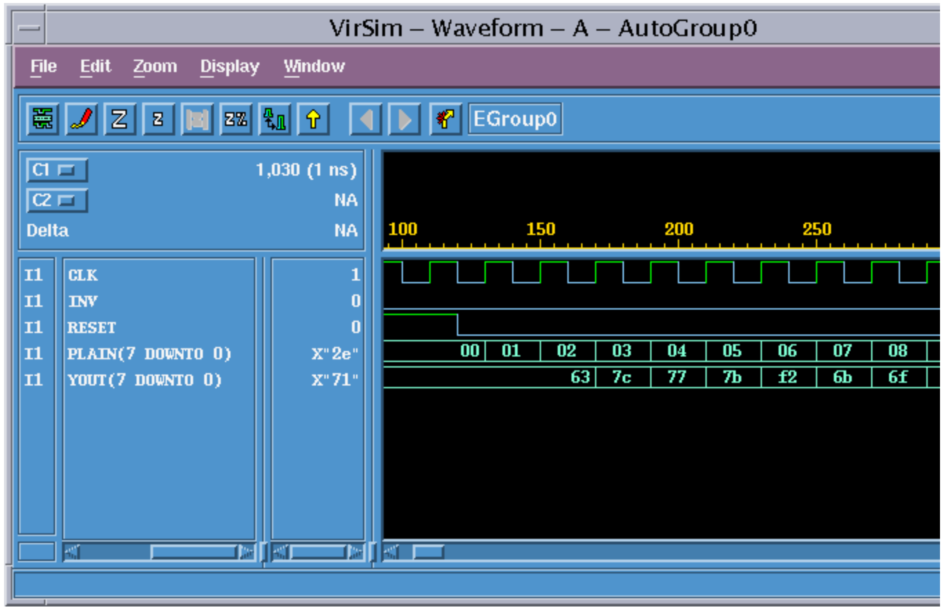
Table 2. Pin list for LEVEL1

SENDER, SUBBYTES (usign ROM, INV function is not supported) and TESTBENCH VHDL file is linked as follow. Please use as it is or modified.

- SENDER: sender.vhd
- SUBBYTES TRANSFORM: subbytes.vhd
- TESTBENCH: test_subbytes.vhd

Figure 4 shows the simulated waveform.

- SENDER generates 8 bit input signal.
- SUBBYTES output is generated with 2 cycle delaye.



```
C1                        1,030 (1 ns)
C2                                  NA
Delta                               NA
I1  CLK                              1
I1  INV                              0
I1  RESET                            0
I1  PLAIN(7 DOWNTO 0)           X"2e"    00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08
I1  YOUT(7 DOWNTO 0)            X"71"              63 | 7c | 77 | 7b | f2 | 6b | 6f
```

Figure 4. Simulation Waveform

---

[6] LEVEL2 Task for experienced designners

In the Level 2 task, you do NOT need to follow the detail design spec.

Mandatory rule is

1) Please design circuit which can perform both SubByes and Inverse SubBytes Transform.

2) Any Architecture, any timing, any pin list are acceptable. I believe you can enjoy your originality.

[7] Speed unit

Since it is impossible to use the same synthesis library for various participants, use the 1 exor gate delay as a unit for speed comparison.

How to measure 1 exor gate delay

1. Synthesize the 50 inputs exor gate
2. Measure the total delay time
3. Unit delay is obtained by total delay divided by the number of stages

- VHDL code for 50 inputs exor : parity.vhd
- example of synthesized circuit : PDF, PS
- example of critical path delay measurement : report_timing
- example of circuit area measurement : report_area

In the previous example, total delay = 7.17 ns and 6 circuit stages, then the 7.17/6= **1.195 ns** is the unit of the speed. Please normalize your circuit speed by this unit.

[8] Report

The report has to include the following contents. Be concise!

Title page	1	Team name, Members Name, School, Grade
	2	Address, Phone, Email-address
	3	T-shirt size for all members in the team
	4	Which level of task is designed.
Contents	1	Circuit block or architecture description
	2	Designed circuit functional explanation, etc.
	3	Appealing point and originality
	4	Critical path speed, and circuit area
	5	HDL codes (VHDL or Verilog HDL)
	6	Simulation waveform indicating the design is operating!
	7	Anything you want to claim

Report has to be emailed to the following address. Please use PDF file format.

If you want to send the report data other than PDF, please consult me.

wada@ie.u-ryukyu.ac.jp

THE DEAD LINE IS 2003/FEBRUARY/6TH!

[9] Suggestion from judges

- We try to evaluate not only the speed and the area, but also your idea ,originality, uniqueness. But be sure to remember that we are not perfect, please make a good presentation to appeal us.
- We definitely take your school grade into account.
- We like fun ideas. Please do something different from others.

[10] Some references

1. Chin-Pin Su et. al, "A Highly Efficient AES Cipher Chip", ASP-DAC2003, pp.561-562, Jan 2003.
2. Satoh A., Morioka, S., Takano, K. and Munetoh, S.: "A Compact Rijndael Hardware Architecture with S-Box Optimization", Advances in Cryptology - ASIACRYPT 2001,LNCS Vol.2248, pp.239-254(2001).
3. University of the Ryukyus LSI Design Contest 2004 Homepage (Information will be updated)
   http://www.ie.u-ryukyu.ac.jp/~wada/design04/contest2004e.html

[11] Acknowledgement

This program is
operated by Univ. of Ryukyus, IE dept.,
co-operated by Okinawa Industry Support Center,
and co-sponsored by SONY LSI Design Inc.

ENJOY HDL! We want to see you at OKINAWA!

Go to Contest Top Page