μ team

SubBytes Circuit Design Submission

(Level 1)

Arif Kusbandono
Ma'muri
Suyanto

bandono@students.ee.itb.ac.id
muri_181@yahoo.com
yanto@ic-proc.paume.itb.ac.id

+62 22 2530001
+62 22 2509713
+62 22 4200502
+62 8562186604
Jl. Aceh no. 88 Bandung
Indonesia
T-Shirt Size **: L**

Complete design documentations are archived in
http://ic.ee.itb.ac.id/~muri/S-Box

undergraduate students of
Electrical Engineering Departement
Institut Teknologi Bandung
Indonesia

Rev. 1.0

Contents

1. SubBytes

1.1. Introduction

The SubBytes is one of the transformation repeatedly used in AES algorithm. The Rijndael algorithm, announced as Advanced Encryption Standards (AES), encrypts information through loops of ShiftRows, SubBytes, MixColumns, and AddRoundKey. Asides from this loops (called round), the key scheduled for each AddRoundKey is generated by also applying SubBytes several times during the key expansion routine [3].

SubsBytes circuit (S-Box) costs outmost in circuits performing AES. Reducing the delay of the S-Box, which is the slowest function block of the entire circuit, as well as reducing its size are then encouraged.

1.2. Architecture Overview

1.2.1. Top Level Circuit Block

The submitted SubBytes transform circuit (S-Box) converts 8 bits data (XIN) to other 8 bits (YOUT). In S-Box mode, the circuit performs a multiplicative inversion over Galois Field $GF(2^8)$ followed by an affine transformation. The inverse transformation is performed in S-Box^{-1} mode of the circuit: inverse affine followed by the same $GF(2^8)$ inversion. A control signal (INV) is then used to switch between these modes.

Table 1-1. Top view pin lists.

SUBBYTES			
signal name	**In/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 S-Box mode, '1' means S-Box^{-1} mode
YOUT	OUT	8	transformed output

Circuit sharing between S-Box and S-Box^{-1} is used provided that both operations need the multiplicative inversion $GF(2^8)$. The circuit block diagram is shown in Figure 1-1. The data path is registered in the $GF(2^8)$ inverter output. Thus, here, we have synchronous inverse $GF(2^8)$ output sampled at CLK rising edge for the next path. During reset this section feeds the next path with "0000 0000". Signals RESET and INV are of asynchronous type.

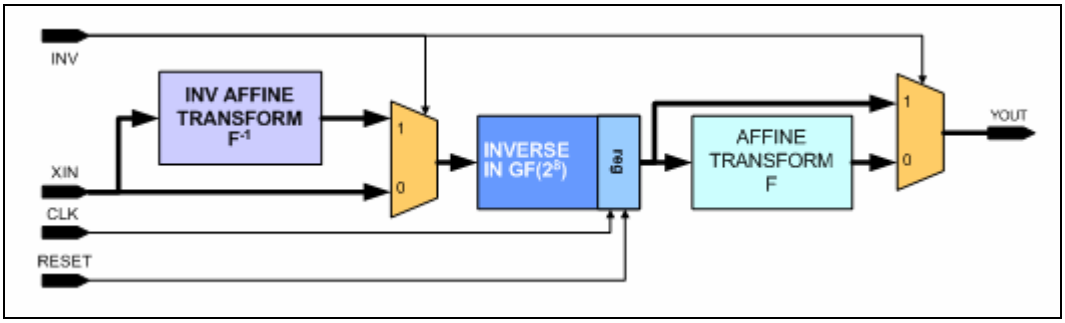


Figure 1-1. Circuit block diagram.

1.2.2. Inverse in GF(2^8)

Inverse $GF(2^8)$ circuit dominates the speed/area cost of the whole S-Box. Various methods for constructing compact inversion circuit were developed e.g._Itoh Tsujii, direct S-Box from look up-table, SOP, POS, BDD, and the composite field technique. The later method effectively reduces these costs through a compact multiplicative inversion circuit. It computes inversion over a field A ($GF(2^8)$), where A, with the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$, is extended from $GF(2)$ (Figure 1-2). The method is sequenced in the following steps [4]:

(1) Mapping of all field A elements to composite field B using an isomorphism function δ.

(2) Computation of the multiplicative inverses over the field B.

(3) Re-mapping of the computation results to A using δ^{-1}.

The term “mapping” is simply reflected through *slicing* of the 8 bits input vector. Slicing allows for choice of composite fields, cheaper (less then 8-bit) operation in smaller field, parallel processing of the slices, conversion between representation in GF, and data rearrangement [1]. Hence, a single degree extension of $GF(2^8)$ will slice the input bits into 4 bits-based computations.

The performance of the data-slicing depends on choosing the right “slice”. Moreover, determination of the irreducible polynomials used (from all possible choices) is done by calculating the complexities e.g.: space complexity (area) is measured by the number of logical XOR and AND. The following irreducibles and constants were chosen in [4] as optimum.

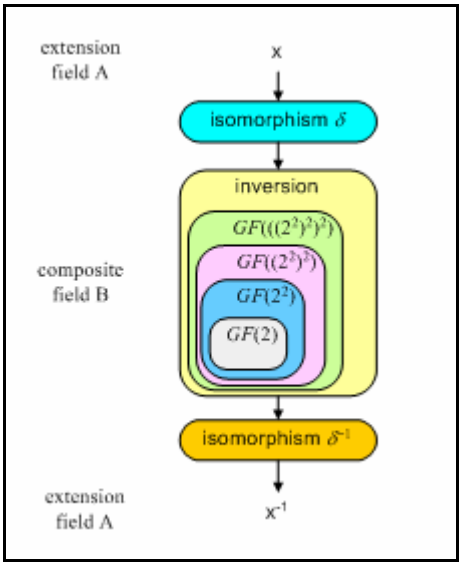


Figure 1-2. The basic idea of working in smaller field.

A single degree extension of $GF(2^8)$ is constructed with the composite field

$$\begin{cases} GF(2^4): m(x) = x^4 + x + 1 \\ GF((2^4)^2): m(x) = x^2 + x + \omega_{14} \end{cases}$$

where $\omega_{14} = [1001]_2$ and $m(x)$ is the corresponding irreducible polynomial, inversion in the extension field $GF(2^8)$ is performed through inversion in the composite field $GF((2^4)^2)$. The submitted design is done using the one that has been developed by Morioka and Satoh [4], from which we have chosen composite field $GF(((2^2)^2)^2)$. The later field is simply obtained by applying multiple extensions of smaller degrees.

Multiple extensions is done in the following:

$$\begin{cases} GF(2^2): m(x) = x^2 + x + 1 \\ GF((2^2)^2): m(x) = x^2 + x + \phi \\ GF(((2^2)^2)^2): m(x) = x^2 + x + \chi \end{cases}$$

where $\phi = [10]_2$, $\chi = [1100]_2$. The correspondent composite field inverter is shown in Figure 1-3.

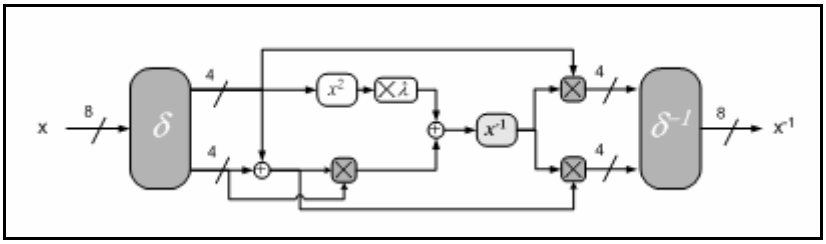


Figure 1-3. The composite field inverter showing 4-bits parallel processing.

Instead of explicitly stating the schematic (Figure 1-3) as separate components in HDL codes, the sub-components of the composite field inverter are converted in PPRM (Positive Polarity Reed-Muller) form: the pre-inversion section, the inversion section, and the post-inversion section [4]. As will be explained further in the following sections, we choose **3-stages PPRM** representation.

1.2.3. Design Project Hierarchy

All design entries were done in HDL Designer Series (FPGA Advantages 5.2). The project hierarchy in the view name is shown in Figure 1-4. The block diagram in Figure 1-1 is not coded as separate components, we use one entity instead.

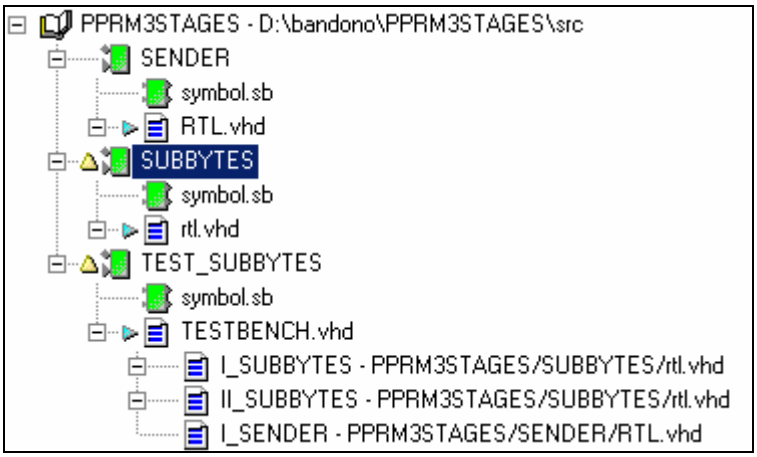


Figure 1-4. HDS browser view of the project library.

1.3. Appealing Point

We claim **originality** through the derivation of the stage representations (Section 2). We have derived back the composite field inverter, the corresponding isomorphism function, and the PPRM representations of the 3-stages. The results are pretty much identical to the ready-to-use versions that were appended in [4].

For evaluation purpose, we've synthesized also other S-Box designs. Since we haven't done proper documentations for each of the other S-Box, short explanations on how each was built are as follow:

- **Look Up-Table**

  We simply modify the subbytes.vhd provided in the Design Contest 2004 homepage to have both SubBytes and InvSubBytes performable. It is basically a ROM.

- **Itoh and Tsujii**

  The Design Contest site also contents example of GF(2^8) inverter. The inversion circuit is based on Itoh Tsujii's algorithm where multiplicative inverse is calculated as $y^{-1} = y^{254}$. Implementation of the later 254 powering consists of multiplication circuit, ^2 (power 2) circuit, ^4 circuit, and ^16 circuit all in $GF(2^8)$. Each of the powering circuit is obtained respectively one after another by deriving the multiplication circuit first. We always perform the necessary AND-XOR minimization during the process.

- **Composite Field GF($(2^4)^2$)**

  Basically, the same design approach as the submitted one is used. Hence, the following sections derivation-method holds. We use the isomorphism functions in [2] and the generic composite field inverter and constant in [4]. Four-bits basis operation is then performed over the subfield $GF(2^4)$.

- **4-stages PPRM**

  The only different to the 3-stages submitted is the staging.

- **SOP and POS**

  The circuit is obtained from its truth table by using two-level logic in terms of *sum of product* and *product of sum*.

1.3.1. Evaluation of Synthesized Circuit

Due to implementation purpose, the design is synthesized into *Xilinx FPGA XCV300PQ240* target library using Xilinx Foundation Series 3.1i . Other types of S-Box design are also synthesized. Comparisons on speed/area criterion are shown in Figure 1-5 and Table 1-2. Complete synthesis report of the submitted design can be found in Section 5.

The delay is normalized using the speed unit of the synthesized 50 input XOR. All synthesis reports are generated under the same constraints. Synthesis is optimized for speed (high effort) under 50 MHz clock constraint and is done using speed grade-6. The absolute values of delay/size can vary depending on the target libraries and synthesis tools, but in [5] the tests were showing almost the same circuit speed ratios.

Table 1-2. Comparison of synthesized circuits using Xilinx target device.

S-Box	*XCV300PQ240*	
	Size (gates)	Delay
Look Up-Table	2662	7.41
Itoh and Tsujii	1960	12.64
Composite Field GF($(2^4)^2$)	682	7.23
3-stages PPRM	***979***	***8.53***
4-stages PPRM	874	7.54
SOP	2410	6.63
POS	2602	7.50

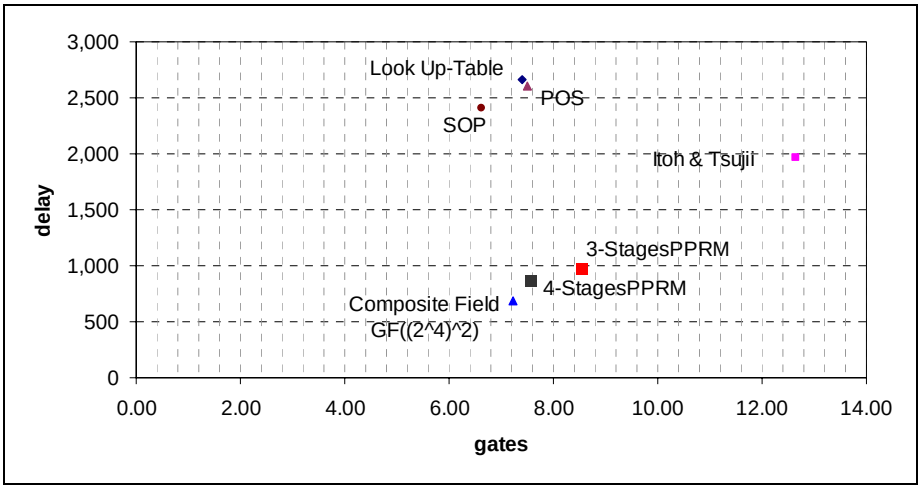


Figure 1-5. Graphical comparison.

1.3.2. Low Power Architecture

Logical AND-gates output '0' (low) for half of the possible input pairs. Due to this fact, logical AND-gates are 50% transparent to dynamic hazard. With similar meaning, XOR are 100% transparent to dynamic hazard (it outputs all the input hazards). Dynamic hazard occurs when different signal arrival times are different between multiple inputs of a gate. Serial connections between multiple gates and some of the internal gates generate hazards. When these hazards propagate along the circuit path, some extra power is consumed. Within the S-Box the gates can switch many times per single transition of the primary inputs.

By organizing AND array that feeds the primary input of XOR array in each stages of PPRM, the hazard transparency of the whole circuit is then reduced. Moreover, arranging in this fashion reduce the power consumption. The composite field inverter arrangement can be shown in Figure 1-6.

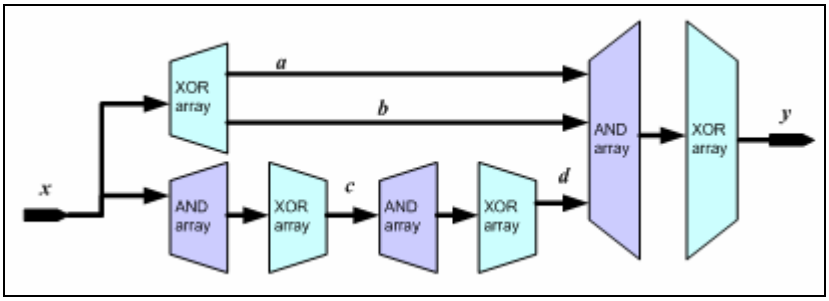


Figure 1-6. Staging in AND-XOR array view.

Full power analysis was done in [4] where in the method, a timing simulation at the gate level were performed using set of tests input data, and the switching activity of all internal gates were then logged. Power was computed from the simulation log and the cell information of the target ASIC library. Being much smaller S-Box, the multi-stage PPRM S-Box achieved the lowest power consumption of 29 μW at 10 MHz using 0.13 μm 1.5V technology. However, the submitted report isn't fully intended for power analysis. We only use the fact that we've arranged the inverter in similar fashion where the 3-stages had been shown in [4] to be the **lowest power consuming S-Box** though it wasn't the smallest n-stages.

2. Derivation of Stage Representations

As mathematical preliminaries in [3], basic operations over the Rijndael field will consist of multiplication, replaced with AND logic, and addition/subtraction, replaced with XOR logic. We derive back the composite field inverter circuit by using the same principals as were explained in [2] for $GF((2^4)^2)$. Then, staging is performed in this $GF(((2^2)^2)^2)$ inverter.

2.1. Derivation of the Composite Field Inversion Formula

The field $GF(2^8)$ can be viewed as $GF(((2^2)^2)^2)$. In both cases we have the same field, we just label the elements differently. We refer to $GF(2^8)$ as the *extension field*. The *extension field irreducible*, $m(x) = x^8 + x^4 + x^3 + x + 1$, defines addition and multiplication over that field. Subsequently, over the *composite field* $GF(((2^2)^2)^2)$, $m(x) = x^2 + x + \lambda$ had been chosen as the *composite field irreducible* in [4]. The coefficient λ is element of $GF((2^2)^2)$. As we continue to attempt computation in smaller field, the *subfield* $GF((2^2)^2)$ is then used. It has the *subfield irreducible* $m(x) = x^2 + x + \phi$, where ϕ is element of $GF(2^2)$. Our next attempt is to derive composite field inverter where we slice 8 bits input into 4 bits parallel processing. This is achievable because we only deal with 16 elements over the subfield $GF((2^2)^2)$.

If we label 256 elements of $GF(2^8)$ with a polynomial of degree 1 whose coefficients are in the subfield $GF((2^2)^2)$, we will have

$$\varsigma \in GF(2^8) \Rightarrow \varsigma = Ax + B$$

where $A, B \in GF((2)^2)^2)$ (the subfield) and x is a root of the composite field irreducible $m(x) = x^2 + x + \lambda = 0$.

Suppose that $u, v \in GF((2^2)^2)^2)$, where $u \bullet v = 1$, that is, $v = u^{-1}$. Through the above labeling we have

$$u = Ax + B$$
$$v = Cx + D$$

where $A, B, C, D \in GF((2^2)^2)$ and x is the same root as before. The multiplication (denoted by $\bullet$) over the composite field $GF(((2^2)^2)^2)$ is performed as

$$u(x) \bullet v(x) = 1$$
$$((Ax + B)(Cx + D)) \bmod (x^2 + x + \lambda) = 1$$
$$(ACx^2 + (AD + BC)x + BD) \bmod (x^2 + x + \lambda) = 1 \qquad (1)$$

Having x as a root of $m(x) = x^2 + x + \lambda = 0$, thus the modulo can be computed using

$$x^2 + x + \lambda = 0$$

$$x^2 = x + \lambda \quad (2)$$

Substituting $x^2 = x + \lambda$ (2)(2) into (1) yields

$$AC(x+\lambda) + (AD + BC)x + BD = 1$$

$$(AC + AD + BC)x + (BD + AC\lambda) = 1$$

The left side polynomial equals to the result of multiplication $u(x) \bullet v(x)$ which represents an element of $GF((2^2)^2)^2)$. Thus, equating will result

$$AC + AD + BC = 0 \quad (3)$$

$$BD + AC\lambda = 1 \quad (4)$$

To find C and D as functions of A and B, we can use both equation (3) and (4) Rearrangement of equation (3) is

$$D = CA^{-1}(A + B) \quad (5)$$

We first find C as a function of A and B by substituting (5) into (4) which yields

$$B(CA^{-1}(A+B)) + AC\lambda = 1$$

$$C(B(A+B) + A^2\lambda) = A$$

$$C = A(AB + B^2 + A^2\lambda)^{-1} \quad (6)$$

D as a function of A and B can be found by substitution of equation (6) into (5). Thus,

$$D = (A+B)(AB + B^2 + A^2\lambda)^{-1} \quad (7)$$

The last two equations ((6) and (7)) can be implemented in hardware using the schematic in Fig.

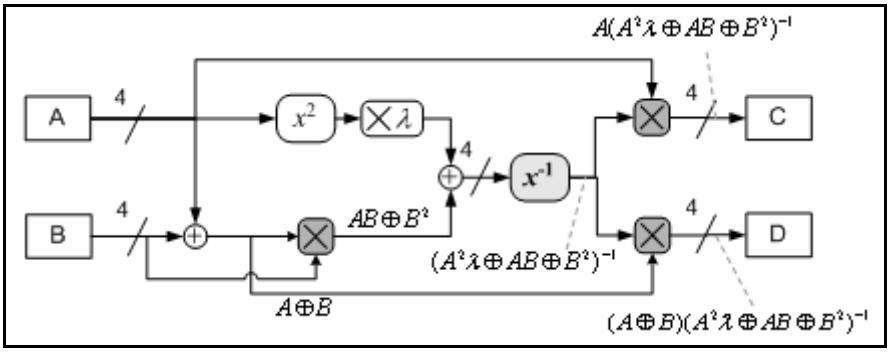


Figure 2-1. The basic composite field inverter.

2.2. Derivation of Isomorphism Function

An eight bit data $\tau = \tau_7\tau_6\tau_5\tau_4\tau_3\tau_2\tau_1\tau_0$ can be expressed as polynomials in Galois Field $GF(2^8)$ in the following representation:

$$\tau(x) = \tau_7.x^7 + \tau_6.x^6 + \tau_5.x^5 + \tau_4.x^4 + \tau_3.x^3 + \tau_2.x^2 + \tau_1.x + \tau_0$$

Similar to the previous section labeling, the above eight bits data in $GF(2^8)$ can be viewed in $GF(((2^2)^2)^2)$ as

$$\tau = a\gamma + b \quad (8)$$

where $a, b \in GF((2^2)^2)$ and γ is a root of the composite field irreducible $S(w)$.

Continuing the extension, any c in $GF((2^2)^2)$ (Galois Field with 16 elements) can be labeled into

$$c = \chi_1\beta + \chi_0 \quad (9)$$

where $\chi_0, \chi_1 \in GF(2^2)$ and β is a root of subfield irreducible $R(z)$. Afterwards, any d in $GF(2^2)$ can be written as

$$d = \delta_1\alpha + \delta_0$$

where $\delta_0, \delta_1 \in GF(2)$ (Galois Field with 2 elements) and α is a root of the subfield irreducible $Q(y)$.

Thus, in terms of α, β, and γ we get (8) extended as

$$\tau = ((a_3\alpha + a_2)\beta + (a_1\alpha + a_0))\gamma + ((b_3\alpha + b_2)\beta + (b_1\alpha + b_0))$$

$$\tau = a_0\gamma + a_1\gamma\alpha + a_2\gamma\beta + a_3\gamma\alpha\beta + b_0 + b_1\alpha + b_2\beta + b_3\alpha\beta$$

stated in vector notation as

$$\vec{\tau} = (1 + \vec{\alpha} + \vec{\beta} + \vec{\alpha}\vec{\beta} \quad \vec{\gamma} + \vec{\gamma}\vec{\alpha} + \vec{\gamma}\vec{\beta} + \vec{\gamma}\vec{\alpha}\vec{\beta}) \begin{pmatrix} \vec{b} \\ \vec{a} \end{pmatrix}$$

where $\vec{\tau} = (\tau_0 \quad \tau_1 \quad \ldots \quad \tau_7)^T$, $\vec{a} = (a_0 \quad a_1 \quad a_2 \quad a_3)^T$, and $\vec{b} = (b_0 \quad b_1 \quad b_2 \quad b_3)^T$.

Note that $\alpha, \beta, \gamma \in GF(2^8)$, so we can write the above equation in the following expanded form:

$$\begin{pmatrix} \tau_0 \\ \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \\ \tau_5 \\ \tau_6 \\ \tau_7 \end{pmatrix} = \begin{pmatrix} 1 & \alpha_0 & \beta_0 & (\alpha\beta)_0 & \gamma_0 & (\gamma\alpha)_0 & (\gamma\beta)_0 & (\gamma\alpha\beta)_0 \\ 0 & \alpha_1 & \beta_1 & (\alpha\beta)_1 & \gamma_1 & (\gamma\alpha)_1 & (\gamma\beta)_1 & (\gamma\alpha\beta)_1 \\ 0 & \alpha_2 & \beta_2 & (\alpha\beta)_2 & \gamma_2 & (\gamma\alpha)_2 & (\gamma\beta)_2 & (\gamma\alpha\beta)_2 \\ 0 & \alpha_3 & \beta_3 & (\alpha\beta)_3 & \gamma_3 & (\gamma\alpha)_3 & (\gamma\beta)_3 & (\gamma\alpha\beta)_3 \\ 0 & \alpha_4 & \beta_4 & (\alpha\beta)_4 & \gamma_4 & (\gamma\alpha)_4 & (\gamma\beta)_4 & (\gamma\alpha\beta)_4 \\ 0 & \alpha_5 & \beta_5 & (\alpha\beta)_5 & \gamma_5 & (\gamma\alpha)_5 & (\gamma\beta)_5 & (\gamma\alpha\beta)_5 \\ 0 & \alpha_6 & \beta_6 & (\alpha\beta)_6 & \gamma_6 & (\gamma\alpha)_6 & (\gamma\beta)_6 & (\gamma\alpha\beta)_6 \\ 0 & \alpha_7 & \beta_7 & (\alpha\beta)_7 & \gamma_7 & (\gamma\alpha)_7 & (\gamma\beta)_7 & (\gamma\alpha\beta)_7 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

It is clear that the matrix above is the isomorphism function δ^{-1}. With $a,b \in GF((2^2)^2)$, we can map the computation result in 4 bits basis (a and b) back to the initial bit-width of τ by using δ^{-1}.

The irreducible polynomials mentioned earlier in this section are

$$\left.\begin{aligned} &GF(2^8): P(x) = x^8 + x^4 + x^3 + x + 1 \\ &GF(2^2): Q(y) = y^2 + y + 1 \\ &GF((2^2)^2): R(z) = z^2 + z + \phi \\ &GF(((2^2)^2)^2): S(w) = w^2 + w + \lambda \end{aligned}\right\} \quad (10)$$

where $\phi = [10]_2$ and $\chi = [1100]_2$. Values of the δ^{-1} matrix elements can be found by solving $Q(\alpha) = 0$, $R(\beta) = 0$, and $S(\gamma) = 0$, respectively:

$$Q(\alpha) = 0$$
$$\alpha^2 + \alpha + 1 = 0$$
$$\alpha^2 + \alpha = 1$$
$$\alpha_6 x^7 + (\alpha_3 \oplus \alpha_5 \oplus \alpha_6)x^6 + \alpha_6 x^5 + (\alpha_2 \oplus \alpha_7)x^4 + (\alpha_3 \oplus \alpha_4 \oplus \alpha_5 \oplus \alpha_6 \oplus \alpha_7)x^3$$
$$+(\alpha_1 \oplus \alpha_2 \oplus \alpha_5)x^2 + (\alpha_1 \oplus \alpha_4 \oplus \alpha_6 \oplus \alpha_7)x + (\alpha_4 \oplus \alpha_6) = 1$$

The solution of the above equation is $\alpha = 1011110\alpha_0$. If we choose $\alpha_0 = 0$, then $\alpha = 10111100$ or in polynomial form $\alpha(x) = x^7 + x^5 + x^4 + x^3 + x^2$.

$$R(\beta) = 0$$
$$\beta^2 + \beta + \phi = 0$$
$$\beta^2 + \beta = \phi = \{10\}_2 = \alpha = x^7 + x^5 + x^4 + x^3 + x^2$$
$$\beta_6 x^7 + (\beta_3 \oplus \beta_5 \oplus \beta_6)x^6 + \beta_6 x^5 + (\beta_2 \oplus \beta_7)x^4 + (\beta_3 \oplus \beta_4 \oplus \beta_5 \oplus \beta_6 \oplus \beta_7)x^3$$
$$+(\beta_1 \oplus \beta_2 \oplus \beta_5)x^2 + (\beta_1 \oplus \beta_4 \oplus \beta_6 \oplus \beta_7)x + (\beta_4 \oplus \beta_6) = x^7 + x^5 + x^4 + x^3 + x^2$$

The solution of the above equation is $\beta = 0101110\beta_0$. If we choose $\beta_0 = 1$, then $\beta = 01011101$ or in polynomial form $\beta(x) = x^6 + x^4 + x^3 + x^2 + 1$.

$$S(\gamma) = 0$$
$$\gamma^2 + \gamma + \lambda = 0$$
$$\gamma^2 + \gamma = \lambda = \{1100\}_2 = (\alpha + 1)\beta = \alpha\beta + \beta = 01010001$$
$$\gamma_6 x^7 + (\gamma_3 \oplus \gamma_5 \oplus \gamma_6)x^6 + \gamma_6 x^5 + (\gamma_2 \oplus \gamma_7)x^4 + (\gamma_3 \oplus \gamma_4 \oplus \gamma_5 \oplus \gamma_6 \oplus \gamma_7)x^3$$
$$+(\gamma_1 \oplus \gamma_2 \oplus \gamma_5)x^2 + (\gamma_1 \oplus \gamma_4 \oplus \gamma_6 \oplus \gamma_7)x + (\gamma_4 \oplus \gamma_6) = x^5 + x^4 + 1$$

The solution of the above equation is $\gamma = 0001111\gamma_0$. If we choose $\gamma_0 = 1$, then $\gamma = 00011111$ or in polynomial form $\gamma(x) = x^5 + x^4 + x^3 + x^2 + x + 1$.

Hence, we have $\alpha = 10111100$, $\beta = 01011101$, and $\gamma = 00011111$. Respectively, $\alpha \bullet \beta = 00001100$, $\gamma \bullet \alpha = 10111011$, $\gamma \bullet \beta = 11110001$, and $\gamma \bullet \alpha \bullet \beta = 10000100$.

Inserting the values into the matrix results in

$$\delta^{-1} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

From the above function, we can find the isomorphism function δ as

$$\delta = (\delta^{-1})^{-1} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

2.3. Derivation of 3-Stages PPRM Representations

The term *three-stages* refers to the amount of partition done for the composite field inverter. For the inverter which we have derived above, the partitioning is shown in Fig. Different staging would cost different speed/area as was shown in [4].

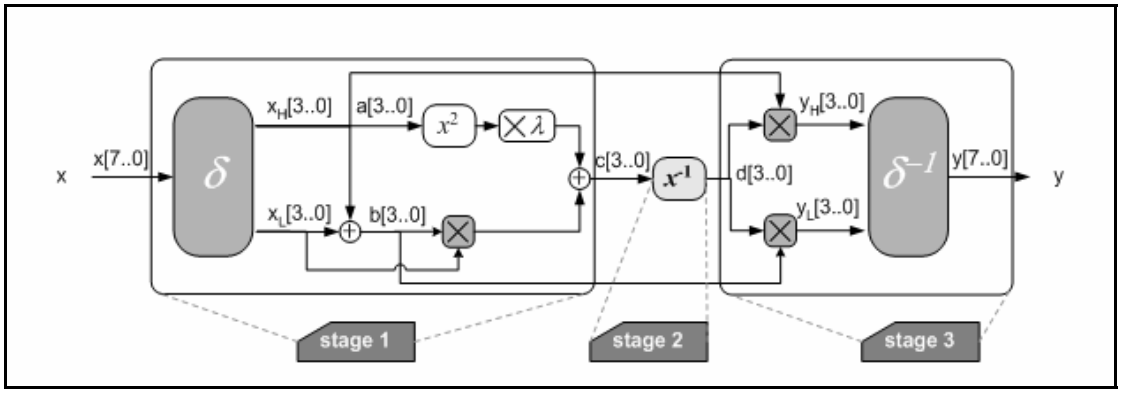


Figure 2-2. Staging of the composite field inverter.

The 8-bits input of the inverter are denoted x and the 8-bits output are denoted y. The other variables a, b, c, and d are the internal wires.

Eight-bits x are sliced into two 4-bits data x_H and x_L through isomorphism function δ. So, we obtain

$$a = x_H \text{ and } b = x_H \oplus x_L \quad (11)$$

In terms of x, x_H, and x_L, the matrix form is

$$\begin{pmatrix} x_H \\ x_L \end{pmatrix} = \delta . x$$

or

$$\begin{pmatrix} x_{L0} \\ x_{L1} \\ x_{L2} \\ x_{L3} \\ x_{H0} \\ x_{H1} \\ x_{H2} \\ x_{H3} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix}$$

Substituting for a and b in (10), we obtain

$$\left.\begin{aligned} a_3 &= x_{H3} = x_7 \oplus x_5 \\ a_2 &= x_{H2} = x_7 \oplus x_6 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\ a_1 &= x_{H1} = x_7 \oplus x_5 \oplus x_3 \oplus x_2 \\ a_0 &= x_{H0} = x_7 \oplus x_5 \oplus x_3 \oplus x_2 \oplus x_1 \\ b_3 &= x_{H3} \oplus x_{L3} = x_6 \oplus x_5 \oplus x_2 \oplus x_1 \\ b_2 &= x_{H2} \oplus x_{L2} = x_6 \\ b_1 &= x_{H1} \oplus x_{L1} = x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\ b_0 &= x_{H0} \oplus x_{L0} = x_7 \oplus x_6 \oplus x_5 \oplus x_3 \oplus x_2 \oplus x_0 \end{aligned}\right\} \quad (12)$$

Through out this section, we are still referring to the list of irreducible polynomials and constants in (10). In the inverter Fig we have the first stage stated

$$c = x_H^2 \lambda \oplus x_H x_L \oplus x_L^2 = a^2 \lambda \oplus ab \oplus b^2 \quad (13)$$

With $m, n \in GF(2^2)$, $m = m_1 m_0$, and $n = n_1 n_0$, the multiplication $p = m \bullet n$ over the field is performed

$$p = m \bullet n = (m_1 \alpha + m_0)(n_1 \alpha + n_0)$$
$$p = m_1 n_1 \alpha^2 + (m_1 n_0 + m_0 n_1)\alpha + m_0 n_0$$

where $m_i, n_i \in GF(2)$ (binary) and $Q(\alpha) = 0$ which gives $\alpha^2 + \alpha + 1 = 0$. Hence, substituting for α^2 yields

$$p = m_1 n_1(\alpha + 1) + (m_1 n_0 + m_0 n_1)\alpha + m_0 n_0$$
$$p = (m_1 n_1 + m_1 n_0 + m_0 n_1)\alpha + (m_1 n_1 + m_0 n_0)$$

With $p = p_1 p_0$, a more hardware-realistic view gives

$$\left.\begin{aligned} p_1 &= (m_1 \cdot n_1 \oplus m_1 \cdot n_0 \oplus m_0 \cdot n_1) \\ p_0 &= (m_1 \cdot n_1 \oplus m_0 \cdot n_0) \end{aligned}\right\} \quad (14)$$

where additions are denoted by $\oplus$ (bitwise XOR) and binary multiplications are denoted by $\cdot$ (AND logic).

Over $GF((2^2)^2)$, multiplication is performed in similar effort. Thus, for $a, b \in GF((2^2)^2)$, we have $q = a \bullet b$ computed

$$q = a \bullet b = (a_H \beta + a_L)(b_H \beta + b_L)$$
$$q = a_H b_H \beta^2 + (a_H b_L + a_L b_H)\beta + a_L b_L$$

where $R(\beta) = \beta^2 + \beta + \phi = 0$ and $a_H, a_L, b_H, b_L \in GF(2^2)$. Hence, substituting for β^2 yields

$$q = a_H b_H(\beta + \phi) + (a_H b_L + a_L b_H)\beta + a_L b_L$$
$$q = (a_H b_H + a_H b_L + a_L b_H)\beta + (a_H b_H \phi + a_L b_L)$$

where $a_H = a_3\alpha + a_2$, $a_L = a_1\alpha + a_0$, $b_H = b_3\alpha + b_2$, $b_L = b_1\alpha + b_0$ and $q = (q_3\alpha + q_2)\beta + (q_1\alpha + q_0)$. All the multiplication terms in $a_i b_i$ above are over $GF(2^2)$. From the previous multiplication over $GF(2^2)$ and having $\phi = [10]_2$, we obtain

$$q_3 = a_3 \cdot b_3 \oplus a_3 \cdot b_2 \oplus a_3 \cdot b_1 \oplus a_3 \cdot b_0 \oplus a_2 \cdot b_3 \oplus a_2 \cdot b_1 \oplus a_1 \cdot b_3 \oplus a_1 \cdot b_2 \oplus a_0 \cdot b_3$$
$$q_2 = a_3 \cdot b_3 \oplus a_3 \cdot b_1 \oplus a_2 \cdot b_2 \oplus a_2 \cdot b_0 \oplus a_1 \cdot b_3 \oplus a_0 \cdot b_2$$
$$q_1 = a_3 \cdot b_2 \oplus a_2 \cdot b_3 \oplus a_2 \cdot b_2 \oplus a_1 \cdot b_1 \oplus a_1 \cdot b_0 \oplus a_0 \cdot b_1$$
$$q_0 = a_3 \cdot b_3 \oplus a_3 \cdot b_2 \oplus a_2 \cdot b_3 \oplus a_1 \cdot b_1 \oplus a_0 \cdot b_0$$

The above derivation leads to the stage representation of (13) where we deal with operation over $GF((2^2)^2)$. For any $r = a^2$ in $GF((2^2)^2)$, we have

$$r_3 = a_3$$
$$r_2 = a_3 \oplus a_2$$
$$r_1 = a_2 \oplus a_1$$
$$r_0 = a_3 \oplus a_1 \oplus a_0$$

Thus, we obtain (13) in

$$\left.\begin{aligned}
c_3 &= a_2 \oplus a_1 \oplus a_0 \oplus b_3 \oplus a_3 \cdot b_3 \oplus a_3 \cdot b_2 \oplus a_3 \cdot b_1 \oplus a_3 \cdot b_0 \oplus a_2 \cdot b_3 \\
&\quad \oplus a_2 \cdot b_1 \oplus a_1 \cdot b_3 \oplus a_1 \cdot b_2 \oplus a_0 \cdot b_3 \\
c_2 &= a_3 \oplus a_0 \oplus b_3 \oplus b_2 \oplus a_3 \cdot b_3 \oplus a_3 \cdot b_1 \oplus a_2 \cdot b_2 \oplus a_2 \cdot b_0 \oplus a_1 \cdot b_3 \\
&\quad \oplus a_0 \cdot b_2 \\
c_1 &= a_3 \oplus b_2 \oplus b_1 \oplus a_3 \cdot b_2 \oplus a_2 \cdot b_3 \oplus a_2 \cdot b_2 \oplus a_1 \cdot b_1 \oplus a_1 \cdot b_0 \oplus a_0 \cdot b_1 \\
c_0 &= a_3 \oplus a_2 \oplus b_3 \oplus b_1 \oplus b_0 \oplus a_3 \cdot b_3 \oplus a_3 \cdot b_2 \oplus a_2 \cdot b_3 \oplus a_1 \cdot b_1 \oplus a_0 \cdot b_0
\end{aligned}\right\} \quad (15)$$

Finally, the stage 1 representation is obtained by substituting (12) into (15).

$$\begin{aligned}
c_3 &= x_7 \cdot x_5 \oplus x_7 \cdot x_4 \oplus x_7 \cdot x_1 \oplus x_7 \cdot x_0 \oplus x_6 \cdot x_5 \oplus x_6 \cdot x_4 \oplus x_6 \cdot x_2 \oplus x_6 \cdot x_1 \\
&\quad \oplus x_5 \cdot x_4 \oplus x_5 \cdot x_2 \oplus x_5 \cdot x_1 \oplus x_5 \cdot x_0 \oplus x_4 \cdot x_2 \oplus x_4 \cdot x_1 \oplus x_3 \cdot x_2 \oplus x_3 \cdot x_1 \oplus x_2 \cdot x_1 \\
c_2 &= x_7 \cdot x_6 \oplus x_7 \cdot x_5 \oplus x_7 \cdot x_3 \oplus x_7 \cdot x_2 \oplus x_7 \cdot x_0 \oplus x_6 \cdot x_5 \oplus x_6 \cdot x_3 \oplus x_6 \cdot x_2 \oplus x_6 \cdot x_1 \\
&\quad \oplus x_6 \cdot x_0 \oplus x_5 \cdot x_3 \oplus x_5 \cdot x_2 \oplus x_4 \cdot x_3 \oplus x_4 \cdot x_2 \oplus x_4 \cdot x_0 \oplus x_3 \cdot x_2 \oplus x_3 \cdot x_0 \\
&\quad \oplus x_2 \cdot x_0 \oplus x_1 \cdot x_0 \\
c_1 &= x_7 \cdot x_5 \oplus x_7 \cdot x_2 \oplus x_7 \cdot x_0 \oplus x_6 \cdot x_5 \oplus x_6 \cdot x_3 \oplus x_5 \cdot x_4 \oplus x_5 \cdot x_3 \oplus x_5 \cdot x_2 \oplus x_5 \cdot x_0 \\
&\quad \oplus x_4 \cdot x_2 \oplus x_3 \cdot x_2 \oplus x_3 \cdot x_0 \oplus x_2 \cdot x_1 \oplus x_2 \cdot x_0 \oplus x_7 \oplus x_5 \oplus x_4 \oplus x_2 \oplus x_1 \\
c_0 &= x_7 \cdot x_6 \oplus x_7 \cdot x_4 \oplus x_7 \cdot x_0 \oplus x_6 \cdot x_5 \oplus x_6 \cdot x_4 \oplus x_6 \cdot x_3 \oplus x_6 \cdot x_1 \oplus x_5 \cdot x_3 \oplus x_5 \cdot x_0 \\
&\quad \oplus x_4 \cdot x_3 \oplus x_4 \cdot x_1 \oplus x_3 \cdot x_2 \oplus x_3 \cdot x_1 \oplus x_3 \cdot x_0 \oplus x_2 \cdot x_0 \oplus x_1 \cdot x_0 \oplus x_6 \oplus x_5 \\
&\quad \oplus x_3 \oplus x_2 \oplus x_0
\end{aligned}$$

In stage 3, we just apply $y_H = a \bullet d$ and $y_L = d \bullet b$, both over the subfield $GF((2^2)^2)$. Applying isomorphism function δ^{-1} as $y = \delta^{-1}(y_L \quad y_H)^T$, then we obtain

$$\begin{aligned}
y_7 &= a_3 \bullet d_0 \oplus a_2 \bullet d_1 \oplus a_2 \bullet d_0 \oplus a_1 \bullet d_2 \oplus a_1 \bullet d_1 \oplus a_1 \bullet d_0 \oplus a_0 \bullet d_3 \oplus a_0 \bullet d_2 \oplus a_0 \bullet d_1 \oplus \\
&\quad d_3 \bullet b_2 \oplus d_2 \bullet b_3 \oplus d_2 \bullet b_2 \oplus d_1 \bullet b_1 \oplus d_1 \bullet b_0 \oplus d_0 \bullet b_1 \\
y_6 &= a_3 \bullet d_3 \oplus a_3 \bullet d_1 \oplus a_2 \bullet d_2 \oplus a_2 \bullet d_0 \oplus a_1 \bullet d_3 \oplus a_0 \bullet d_2 \oplus d_3 \bullet b_3 \oplus d_3 \bullet b_1 \oplus d_2 \bullet b_2 \oplus \\
&\quad d_2 \bullet b_0 \oplus d_1 \bullet b_3 \oplus d_0 \bullet b_2 \\
y_5 &= a_3 \bullet d_3 \oplus a_3 \bullet d_2 \oplus a_3 \bullet d_1 \oplus a_2 \bullet d_3 \oplus a_2 \bullet d_0 \oplus a_1 \bullet d_3 \oplus a_1 \bullet d_1 \oplus a_1 \bullet d_0 \oplus a_0 \bullet d_2 \oplus \\
&\quad a_0 \bullet d_1 \oplus d_3 \bullet b_2 \oplus d_2 \bullet b_3 \oplus d_2 \bullet b_2 \oplus d_1 \bullet b_1 \oplus d_1 \bullet b_0 \oplus d_0 \bullet b_1 \\
y_4 &= a_3 \bullet d_1 \oplus a_2 \bullet d_0 \oplus a_1 \bullet d_3 \oplus a_1 \bullet d_0 \oplus a_0 \bullet d_2 \oplus a_0 \bullet d_1 \oplus a_0 \bullet d_0 \oplus d_3 \bullet b_3 \oplus d_3 \bullet b_2 \oplus \\
&\quad d_3 \bullet b_1 \oplus d_2 \bullet b_3 \oplus d_2 \bullet b_0 \oplus d_1 \bullet b_3 \oplus d_1 \bullet b_1 \oplus d_1 \bullet b_0 \oplus d_0 \bullet b_2 \oplus d_0 \bullet b_1 \\
y_3 &= a_3 \bullet d_3 \oplus a_2 \bullet d_2 \oplus a_1 \bullet d_0 \oplus a_0 \bullet d_1 \oplus a_0 \bullet d_0 \oplus d_3 \bullet b_0 \oplus d_2 \bullet b_1 \oplus d_2 \bullet b_0 \oplus d_1 \bullet b_2 \oplus \\
&\quad d_1 \bullet b_1 \oplus d_1 \bullet b_0 \oplus d_0 \bullet b_3 \oplus d_0 \bullet b_2 \oplus d_0 \bullet b_1 \\
y_2 &= a_3 \bullet d_1 \oplus a_3 \bullet d_0 \oplus a_2 \bullet d_1 \oplus a_1 \bullet d_3 \oplus a_1 \bullet d_2 \oplus a_1 \bullet d_1 \oplus a_0 \bullet d_3 \oplus a_0 \bullet d_0 \oplus d_3 \bullet b_0 \oplus \\
&\quad d_2 \bullet b_1 \oplus d_2 \bullet b_0 \oplus d_1 \bullet b_2 \oplus d_1 \bullet b_1 \oplus d_1 \bullet b_0 \oplus d_0 \bullet b_3 \oplus d_0 \bullet b_2 \oplus d_0 \bullet b_1 \\
y_1 &= a_3 \bullet d_3 \oplus a_2 \bullet d_2 \oplus a_1 \bullet d_0 \oplus a_0 \bullet d_1 \oplus a_0 \bullet d_0 \\
y_0 &= a_3 \bullet d_1 \oplus a_2 \bullet d_0 \oplus a_1 \bullet d_3 \oplus a_1 \bullet d_0 \oplus a_0 \bullet d_2 \oplus a_0 \bullet d_1 \oplus a_0 \bullet d_0 \oplus d_3 \bullet b_2 \oplus d_3 \bullet b_1 \oplus \\
&\quad d_2 \bullet b_3 \oplus d_2 \bullet b_2 \oplus d_2 \bullet b_0 \oplus d_1 \bullet b_3 \oplus d_1 \bullet b_1 \oplus d_0 \bullet b_2 \oplus d_0 \bullet b_0
\end{aligned}$$

Stage 2 of the composite field inverter is an inverter as well. It computes multiplicative inverse over the subfield $GF((2^2)^2)$. The structure derived in Section 2.1 still holds for the

subfield inverter. We only change the labeling of the 256 elements of $GF(((2^2)^2)^2)$. Alternatively, all 16 elements of the subfield $GF((2^2)^2)$ are labeled in the same manner as done in (9). By changing all the label coefficients and the irreducible polynomial constant with elements of $GF(2^2)$, we get the schematic in fig.

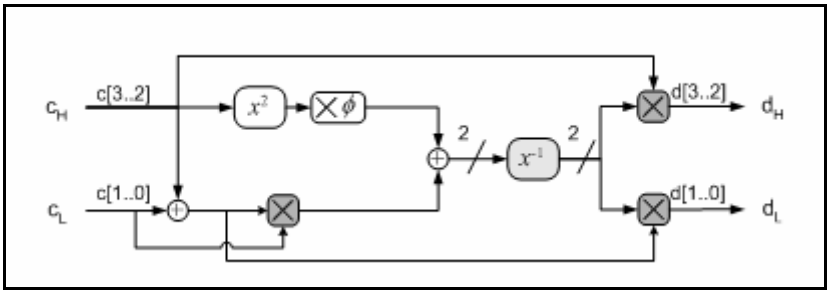
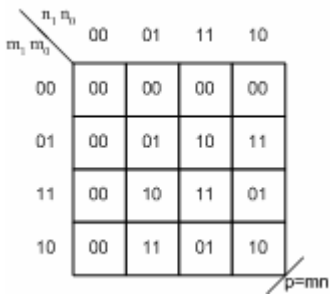


Figure 2-3. Stage 2, the subfield GF((2^2)^2) inverter.

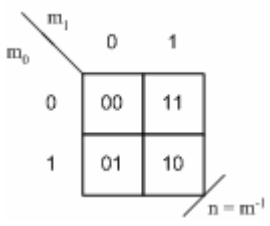
From fig, we can express d as function of c which gives

$$\left.\begin{aligned} d_H &= c_H(c_H^2\phi + c_H c_L + c_L^2)^{-1} \\ d_L &= (c_H + c_L)(c_H^2\phi + c_H c_L + c_L^2)^{-1} \end{aligned}\right\} \qquad (16)$$

where $d_H = d_3\alpha + d_2$, $d_L = d_1\alpha + d_0$, $c_H = c_3\alpha + c_2$, and $c_H = c_1\alpha + c_0$. Since the computations are over $GF(2^2)$, we can reuse the multiplication in (14), where applying all possible inputs yields



Now, our objective is to derive the inverter for the subfield $GF(2^2)$ which is a component of the $GF((2^2)^2)$ inverter (Figure 2-3). Rearrangement of the previous K-map gives



Thus, for the $n = m^{-1}$ K-map, minimization results are $n_1 = m_1$ and $n_0 = m_1 \oplus m_0$. By applying the later multiplicative inverse and operations over $GF(2^2)$ to (16), we obtain the stage 2 representation as

$$d_3 = (c_3 \bullet c_2 \bullet c_1) \oplus (c_3 \bullet c_0) \oplus c_3 \oplus c_2$$
$$d_2 = (c_3 \bullet c_2 \bullet c_1) \oplus (c_3 \bullet c_2 \bullet c_0) \oplus (c_3 \bullet c_0) \oplus (c_2 \bullet c_1) \oplus c_2$$
$$d_1 = (c_3 \bullet c_2 \bullet c_1) \oplus (c_3 \bullet c_1 \bullet c_0) \oplus (c_2 \bullet c_0) \oplus c_3 \oplus c_2 \oplus c_1$$
$$d_0 = (c_3 \bullet c_2 \bullet c_1) \oplus (c_3 \bullet c_2 \bullet c_0) \oplus (c_3 \bullet c_1 \bullet c_0) \oplus (c_3 \bullet c_1) \oplus (c_3 \bullet c_0) \oplus (c_2 \bullet c_1 \bullet c_0) \oplus (c_2 \bullet c_1) \oplus c_2 \oplus c_1 \oplus c_0$$

3. VHDL Listings

3.1. The SubBytes HDL

```
 --
 -- File Name : SubBytes.vhd
 -- Function : combinational logic of SubBytes transform using pprm 3
stages
 --
 --
 -- Created by - Micro Team
 --

 LIBRARY ieee;
 USE ieee.std_logic_1164.all;
 USE ieee.numeric_std.all;


 ENTITY SubBytes IS
 -- Declarations
  port (CLK    : in  std_logic;
           RESET  : in  std_logic;
           XIN    : in  unsigned( 7 downto 0);
           INV    : in  std_logic;
           YOUT   : out unsigned( 7 downto 0)
    );


 END SubBytes ;

 -- hds interface_end
 ARCHITECTURE rtl OF SubBytes IS

 signal invin : unsigned(7 downto 0);
 signal invout : unsigned(7 downto 0);


 Function Inverse_GF(X: unsigned(7 downto 0)) Return unsigned IS

  Variable A : unsigned(3 downto 0);
    Variable B : unsigned(3 downto 0);
    Variable C : unsigned(3 downto 0);
    Variable D : unsigned(3 downto 0);
  Variable Y : unsigned(7 downto 0);

 Begin

 -- STAGE 1 --

 --Define A--
 A(3) := X(7) xor X(5);
 A(2) := X(7) xor X(6) xor X(4) xor X(3) xor X(2) xor X(1);
 A(1) := X(7) xor X(5) xor X(3) xor X(2);
 A(0) := X(7) xor X(5) xor X(3) xor X(2) xor X(1);

 --Define B--
 B(3) := X(6) xor X(5) xor X(2) xor X(1);
 B(2) := X(6);
```

```
 B(1) := X(7) xor X(6) xor X(5) xor X(4) xor X(3) xor X(2) xor X(1);
 B(0) := X(7) xor X(6) xor X(5) xor X(3) xor X(2) xor X(0);

 --Define C--
 C(3) := (X(7) and X(5)) xor (X(7) and X(4)) xor (X(7) and X(1)) xor
(X(7) and X(0)) xor (X(6) and X(5)) xor (X(6) and X(4))
  xor (X(6) and X(2)) xor (X(6) and X(1)) xor (X(5) and X(4)) xor
(X(5) and X(2)) xor (X(5) and X(1)) xor (X(5) and X(0))
  xor (X(4) and X(2)) xor (X(4) and X(1)) xor (X(3) and X(2)) xor
(X(3) and X(1)) xor (X(2) and X(1)) ;

 C(2) := (X(7) and X(6)) xor (X(7) and X(5)) xor (X(7) and X(3)) xor
(X(7) and X(2)) xor (X(7) and X(0)) xor (X(6) and X(5))
  xor (X(6) and X(3)) xor (X(6) and X(2)) xor (X(6) and X(1)) xor
(X(6) and X(0)) xor (X(5) and X(3)) xor (X(5) and X(2))
  xor (X(4) and X(3)) xor (X(4) and X(2)) xor (X(4) and X(0)) xor
(X(3) and X(2)) xor (X(3) and X(0)) xor (X(2) and X(0))
  xor (X(1) and X(0));

 C(1) := (X(7) and X(5)) xor (X(7) and X(2)) xor (X(7) and X(0)) xor
(X(6) and X(5)) xor (X(6) and X(3)) xor (X(5) and X(4))
  xor (X(5) and X(3)) xor (X(5) and X(2)) xor (X(5) and X(0)) xor
(X(4) and X(2)) xor (X(3) and X(2)) xor (X(3) and X(0))
  xor (X(2) and X(1)) xor (X(2) and X(0)) xor X(7) xor X(5) xor X(4)
xor X(2) xor X(1);

 C(0) := (X(7) and X(6)) xor (X(7) and X(4)) xor (X(7) and X(0)) xor
(X(6) and X(5)) xor (X(6) and X(4)) xor (X(6) and X(3))
  xor (X(6) and X(1)) xor (X(5) and X(3)) xor (X(5) and X(0)) xor
(X(4) and X(3)) xor (X(4) and X(1)) xor (X(3) and X(2))
  xor (X(3) and X(1)) xor (X(3) and X(0)) xor (X(2) and X(0)) xor
(X(1) and X(0))
  xor X(6) xor X(5) xor X(3) xor X(2) xor X(0);


 -- STAGE 2 --

 --Define D--
 D(3) := (C(3) and C(2) and C(1)) xor (C(3) and C(0)) xor C(3) xor
C(2);
 D(2) := (C(3) and C(2) and C(1)) xor (C(3) and C(2) and C(0)) xor
(C(3) and C(0)) xor (C(2) and C(1)) xor C(2);
 D(1) := (C(3) and C(2) and C(1)) xor (C(3) and C(1) and C(0)) xor
(C(2) and C(0)) xor C(3) xor C(2) xor C(1);
 D(0) := (C(3) and C(2) and C(1)) xor (C(3) and C(2) and C(0)) xor
(C(3) and C(1) and C(0)) xor (C(3) and C(1))
  xor (C(3)and C(0)) xor (C(2) and C(1) and C(0)) xor (C(2) and C(1))
xor C(2) xor C(1) xor C(0);


 -- STAGE 3 --

 --Define Y--
 Y(7) := (A(3) and D(0)) xor (A(2) and D(1)) xor (A(2) and D(0)) xor
(A(1) and D(2)) xor (A(1) and D(1))
  xor (A(1) and D(0)) xor (A(0) and D(3)) xor (A(0) and D(2)) xor
(A(0) and D(1)) xor (D(3) and B(2))
  xor (D(2) and B(3)) xor (D(2) and B(2)) xor (D(1) and B(1)) xor
(D(1) and B(0)) xor (D(0) and B(1));
```

```
 Y(6) := (A(3) and D(3)) xor (A(3) and D(1)) xor (A(2) and D(2)) xor
(A(2) and D(0)) xor (A(1) and D(3))
  xor (A(0) and D(2)) xor (D(3) and B(3)) xor (D(3) and B(1)) xor
(D(2) and B(2)) xor (D(2) and B(0))
  xor (D(1) and B(3)) xor (D(0) and B(2));

 Y(5) := (A(3) and D(3)) xor (A(3) and D(2)) xor (A(3) and D(1)) xor
(A(2) and D(3)) xor (A(2) and D(0))
  xor (A(1) and D(3)) xor (A(1) and D(1)) xor (A(1) and D(0)) xor
(A(0) and D(2)) xor (A(0) and D(1))
  xor (D(3) and B(2)) xor (D(2) and B(3)) xor (D(2) and B(2)) xor
(D(1) and B(1)) xor (D(1) and B(0))
  xor (D(0) and B(1));

 Y(4) := (A(3) and D(1)) xor (A(2) and D(0)) xor (A(1) and D(3)) xor
(A(1) and D(0)) xor (A(0) and D(2))
  xor (A(0) and D(1)) xor (A(0) and D(0)) xor (D(3) and B(3)) xor
(D(3) and B(2)) xor (D(3) and B(1))
  xor (D(2) and B(3)) xor (D(2) and B(0)) xor (D(1) and B(3)) xor
(D(1) and B(1)) xor (D(1) and B(0))
  xor (D(0) and B(2)) xor (D(0) and B(1));

 Y(3) := (A(3) and D(3)) xor (A(2) and D(2)) xor (A(1) and D(0)) xor
(A(0) and D(1)) xor (A(0) and D(0))
  xor (D(3) and B(0)) xor (D(2) and B(1)) xor (D(2) and B(0)) xor
(D(1) and B(2)) xor (D(1) and B(1))
  xor (D(1) and B(0)) xor (D(0) and B(3)) xor (D(0) and B(2)) xor
(D(0) and B(1));

 Y(2) := (A(3) and D(1)) xor (A(3) and D(0)) xor (A(2) and D(1)) xor
(A(1) and D(3)) xor (A(1) and D(2))
  xor (A(1) and D(1)) xor (A(0) and D(3)) xor (A(0) and D(0)) xor
(D(3) and B(0)) xor (D(2) and B(1))
  xor (D(2) and B(0)) xor (D(1) and B(2)) xor (D(1) and B(1)) xor
(D(1) and B(0)) xor (D(0) and B(3))
  xor (D(0) and B(2)) xor (D(0) and B(1));

 Y(1) := (A(3) and D(3)) xor (A(2) and D(2)) xor (A(1) and D(0)) xor
(A(0) and D(1)) xor (A(0) and D(0));

 Y(0) := (A(3) and D(1)) xor (A(2) and D(0)) xor (A(1) and D(3)) xor
(A(1) and D(0)) xor (A(0) and D(2))
  xor (A(0) and D(1)) xor (A(0) and D(0)) xor (D(3) and B(2)) xor
(D(3) and B(1)) xor (D(2) and B(3))
  xor (D(2) and B(2)) xor (D(2) and B(0)) xor (D(1) and B(3)) xor
(D(1) and B(1)) xor (D(0) and B(2))
  xor (D(0) and B(0));


 return Y;
 end inverse_GF;


 BEGIN

 INVERSE: process(CLK)
 begin
     if rising_edge(CLK) then
         if (RESET='1') then
             invout <= "00000000";
```

```
        else
            invout <= Inverse_GF(invin);
        end if;
    end if;
end process INVERSE;

process (inv,xin,invout)
begin

 if(inv='0')then
 invin <= xin;

 yout(7) <= invout(7) xor invout(6) xor invout(5) xor invout(4) xor
invout(3) xor '0';
 yout(6) <= invout(6) xor invout(5) xor invout(4) xor invout(3) xor
invout(2) xor '1';
 yout(5) <= invout(5) xor invout(4) xor invout(3) xor invout(2) xor
invout(1) xor '1';
 yout(4) <= invout(4) xor invout(3) xor invout(2) xor invout(1) xor
invout(0) xor '0';
 yout(3) <= invout(7) xor invout(3) xor invout(2) xor invout(1) xor
invout(0) xor '0';
 yout(2) <= invout(7) xor invout(6) xor invout(2) xor invout(1) xor
invout(0) xor '0';
 yout(1) <= invout(7) xor invout(6) xor invout(5) xor invout(1) xor
invout(0) xor '1';
 yout(0) <= invout(7) xor invout(6) xor invout(5) xor invout(4) xor
invout(0) xor '1';
 else
 invin(7) <= xin(6) xor xin(4) xor xin(1) xor '0';
 invin(6) <= xin(5) xor xin(3) xor xin(0) xor '0';
 invin(5) <= xin(7) xor xin(4) xor xin(2) xor '0';
 invin(4) <= xin(6) xor xin(3) xor xin(1) xor '0';
 invin(3) <= xin(5) xor xin(2) xor xin(0) xor '0';
 invin(2) <= xin(7) xor xin(4) xor xin(1) xor '1';
 invin(1) <= xin(6) xor xin(3) xor xin(0) xor '0';
 invin(0) <= xin(7) xor xin(5) xor xin(2) xor '1';

 yout <= invout;

 end if;
end process;

END rtl;
```

3.2. Testbench HDL

```
-- Univ. of the Ryukyus LSI design contest 2004
-- SubBytes Transform Circuit for AES Cipher
-- file: test_subbytes.vhd
-- TESTBENCH
-- Tom Wada 2003/September/15
-- Modified by Micro Team

library IEEE;
use IEEE.STD_LOGIC_1164.all, IEEE.NUMERIC_STD.all;

entity TEST_SUBBYTES is
```

```
end TEST_SUBBYTES;

architecture TESTBENCH of TEST_SUBBYTES is
  -- sender
  component SENDER
      port ( CLK    : in  std_logic;
             RESET  : in  std_logic;
             PLAIN  : out unsigned (7 downto 0) );
  end component;
  -- subbytes
  component SUBBYTES
      port ( CLK    : in  std_logic;
          RESET  : in  std_logic;
        XIN    : in  unsigned(7 downto 0);
             INV    : in  std_logic;
             YOUT   : out unsigned(7 downto 0) );
  end component;

-- system clock
signal  CLK  : std_logic  := '0' ;
-- system reset
signal RESET : std_logic  := '1';
-- cycle count
signal cycle : integer :=0;
-- wires on the board
signal PLAIN,PLAIN2  : unsigned (7 downto 0);
signal YOUT   : unsigned (7 downto 0);
signal INV    : std_logic  := '0';
signal NOTINV : std_logic  := '1';

begin

-- clock generator
    CLOCK_GEN: process
    begin
        if (cycle < 1000) then
            cycle <= cycle + 1;
            wait for 10 ns;
            CLK <= not CLK;
        else wait;
        end if;
    end process CLOCK_GEN;

-- reset sequence
    RESET_GEN: process
    begin
        LOOP1: for N in 0 to 5 loop
           wait until falling_edge(CLK);
        end loop LOOP1;
        RESET <= '0';
    end process RESET_GEN;

-- sender instance
    I_SENDER: SENDER port map(CLK,RESET,PLAIN);

-- subbytes instance
    I_SUBBYTES: SUBBYTES port map(CLK,RESET,PLAIN,INV,YOUT);
 II_SUBBYTES: SUBBYTES port map(CLK,RESET,YOUT,NOTINV,PLAIN2);

end TESTBENCH;
```

3.3. The Sender HDL

```
-- hds header_start
-- hds header_end
-- Univ. of the Ryukyus LSI design contest 2004
-- SubBytes Transform Circuit for AES Cipher
-- file: sender.vhd
-- Sender generates 8 bit integer from 0 to 255
-- Tom Wada 2003/September/15

library IEEE;
use IEEE.STD_LOGIC_1164.all, IEEE.NUMERIC_STD.all;

entity SENDER is
    port ( CLK    : in  std_logic;
           RESET  : in  std_logic;
           PLAIN  : out unsigned (7 downto 0) );
end SENDER;

-- hds interface_end


architecture RTL of SENDER is
-- signal define
signal  count : unsigned (7 downto 0); -- 8 bit counter

begin
-------------------------------------------
--  8 bit counter
-------------------------------------------
COUNTER: process(CLK)
begin
    if rising_edge(CLK) then
        if (RESET='1') then
            count <= "00000000";
        else
            count <= count + 1;
        end if;
    end if;
end process COUNTER;

-------------------------------------------
--  OUTPUT GEN
-------------------------------------------
PLAIN <= count;

end RTL;
```

4. Simulation

4.1. Testbench Circuit

Simulation is done using the provided testbench, sender.vhd. The SubBytes simulation result is verified using the S-Box table given in [3]. We slightly modify the testbench to have both SubByte and InvSubByte simulated.

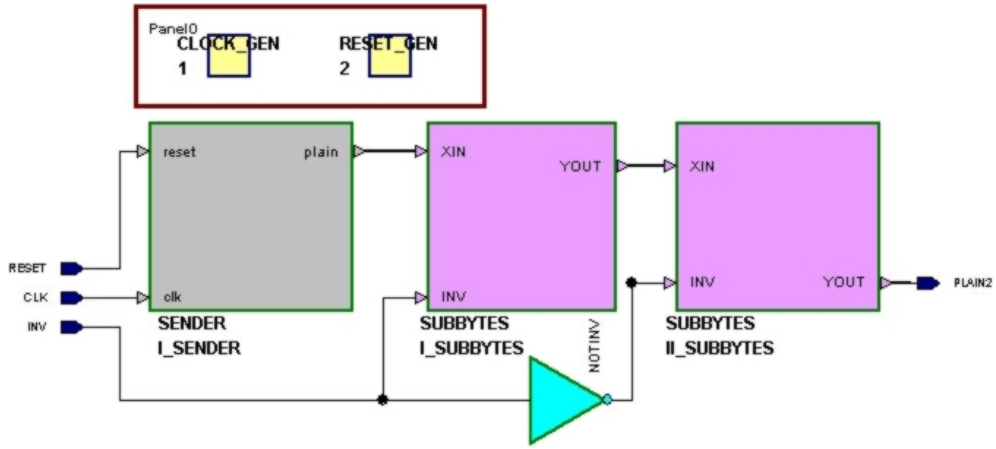
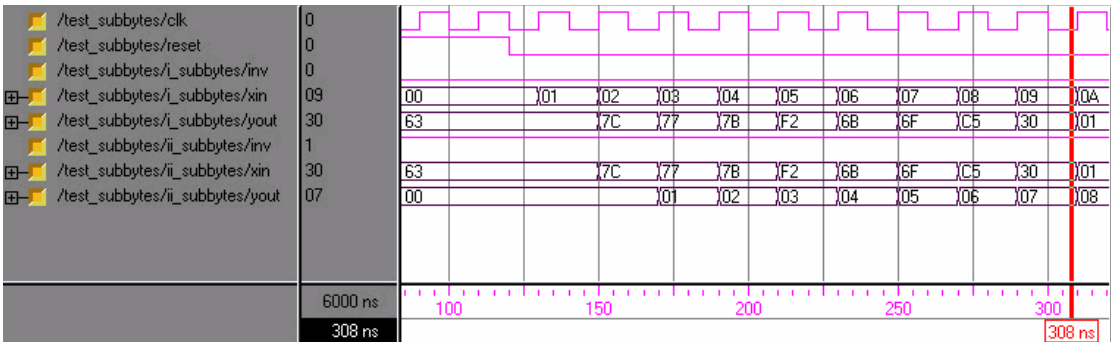


Figure 4-1. Testbench circuit for both S-Box and S-Box^{-1} mode.

4.2. Waveform

The figure below is a screen capture of the simulated waveform. The complete simulation waveform can be found in the Appendix. The timing specification differs from the Level 1 requirement. This is because we only have the data path registered once. However the throughput is still the same.



5. Critical Path and Circuit Area

5.1. Timing Report of *parity.vhd*

```
================================================================================
Timing constraint: Default path analysis
 50 items analyzed, 0 timing errors detected.
 Maximum delay is  15.997ns.
--------------------------------------------------------------------------------
Delay:    15.997ns A<3> to Y

Path A<3> to Y contains 5 levels of logic:
Path starting from Comp: P67.PAD
To                    Delay type         Delay(ns)  Physical Resource
                                                    Logical
Resource(s)
-------------------------------------------------  --------
P67.I                 Tiopi                0.768R  A<3>
                                                    A<3>.PAD
                                                    C_A<3>
CLB_R19C1.S0.F2       net (fanout=1)       3.908R  N_A<3>
CLB_R19C1.S0.X        Topx                 0.727R  BLK2_2
                                                    C56
                                                    C55
CLB_R24C1.S1.F4       net (fanout=1)       1.304R  syn1571
CLB_R24C1.S1.X        Topx                 0.727R  BLK1_2
                                                    C54
                                                    C53
CLB_R20C1.S0.F1       net (fanout=1)       1.562R  syn1574
CLB_R20C1.S0.X        Topx                 0.727R  BLK0_2
                                                    C52
                                                    C51
P47.O                 net (fanout=1)       1.487R  N_Y
P47.PAD               Tioop                4.787R  Y
                                                    C_Y
                                                    Y.PAD
--------------------------------------------------
Total (7.736ns logic, 8.261ns route)      15.997ns
      (48.4% logic, 51.6% route)

--------------------------------------------------------------------------------
```

5.2. SubBytes Reports

5.2.1. Area

```
Design Summary
--------------
   Number of errors:      0
   Number of warnings:   32
   Number of Slices:                 71 out of  3,072    2%
   Number of Slices containing
      unrelated logic:                0 out of     71    0%
   Number of Slice Flip Flops:        8 out of  6,144    1%
   Number of 4 input LUTs:          137 out of  6,144    2%
   Number of bonded IOBs:            18 out of    166   10%
   Number of GCLKs:                   1 out of      4   25%
   Number of GCLKIOBs:                1 out of      4   25%
Total equivalent gate count for design:  979
Additional JTAG gate count for IOBs:  912
```

5.2.2. Timing

```
Xilinx TRACE, Version D.19
Copyright (c) 1995-2000 Xilinx, Inc.  All rights reserved.

trce subbytes.ncd subbytes.pcf -e 3 -o subbytes.twr

Design file:              subbytes.ncd
Physical constraint file: subbytes.pcf
Device,speed:             xcv300,-6 (PRELIMINARY 1.102 2000-05-03)
Report level:             error report
------------------------------------------------------------------------
----------

WARNING:Timing:2491 - No timing constraints found, doing default
enumeration.

================================================================================
===========
Timing constraint: Default period analysis
 65409 items analyzed, 0 timing errors detected.
 Minimum period is  27.305ns.
------------------------------------------------------------------------
----------

================================================================================
===========
Timing constraint: Default net enumeration
 149 items analyzed, 0 timing errors detected.
 Maximum net delay is   5.144ns.
------------------------------------------------------------------------
----------


All constraints were met.


Data Sheet report:
-----------------
All values displayed in nanoseconds (ns)

Setup/Hold to clock CLK
---------------+------------+------------+
               |  Setup to  |  Hold to   |
Source Pad     | clk (edge) | clk (edge) |
---------------+------------+------------+
INV            |   25.531(R)|    0.000(R)|
RESET          |    4.748(R)|    0.000(R)|
XIN<0>         |   22.402(R)|    0.000(R)|
XIN<1>         |   25.379(R)|    0.000(R)|
XIN<2>         |   25.358(R)|    0.000(R)|
XIN<3>         |   22.300(R)|    0.000(R)|
XIN<4>         |   24.932(R)|    0.000(R)|
XIN<5>         |   24.525(R)|    0.000(R)|
XIN<6>         |   24.727(R)|    0.000(R)|
XIN<7>         |   23.715(R)|    0.000(R)|
---------------+------------+------------+

Clock CLK to Pad
---------------+------------+
```

```
               | clk (edge) |
Destination Pad|   to PAD   |
---------------+------------+
YOUT<0>        |   14.607(R)|
YOUT<1>        |   13.583(R)|
YOUT<2>        |   13.969(R)|
YOUT<3>        |   13.679(R)|
YOUT<4>        |   13.580(R)|
YOUT<5>        |   14.466(R)|
YOUT<6>        |   14.420(R)|
YOUT<7>        |   14.053(R)|
---------------+------------+

Pad to Pad
---------------+---------------+---------+
Source Pad     |Destination Pad|  Delay  |
---------------+---------------+---------+
INV            |YOUT<0>        |   13.226|
INV            |YOUT<1>        |   11.941|
INV            |YOUT<2>        |   12.370|
INV            |YOUT<3>        |   12.676|
INV            |YOUT<4>        |   13.359|
INV            |YOUT<5>        |   13.702|
INV            |YOUT<6>        |   13.310|
INV            |YOUT<7>        |   13.520|
---------------+---------------+---------+


Timing summary:
---------------

Timing errors: 0  Score: 0

Constraints cover 65409 paths, 149 nets, and 505 connections (100.0%
coverage)

Design statistics:
   Minimum period:  27.305ns (Maximum frequency:  36.623MHz)
   Maximum net delay:   5.144ns


Analysis completed Sun Feb 01 16:55:32 2004
```

5.3. Synthesis Report Summaries

From parity.vhd timing report (Sec. 5.1), the total critical path delay is 15.997ns within 5 circuit stages. Hence, unit delay = 15.997ns/5 = **3.199ns**. Therefore, our design speed equals to 27.305/3.199 = **8.53** unit delay. The area of the SubBytes reported is equivalent to 979 gates.

6. References

[1] A. Rudra et al, “Efficient Galois Field Arithmetic on SIMD Architectures”, http://www.cs.utexas.edu/users/atri/papers.

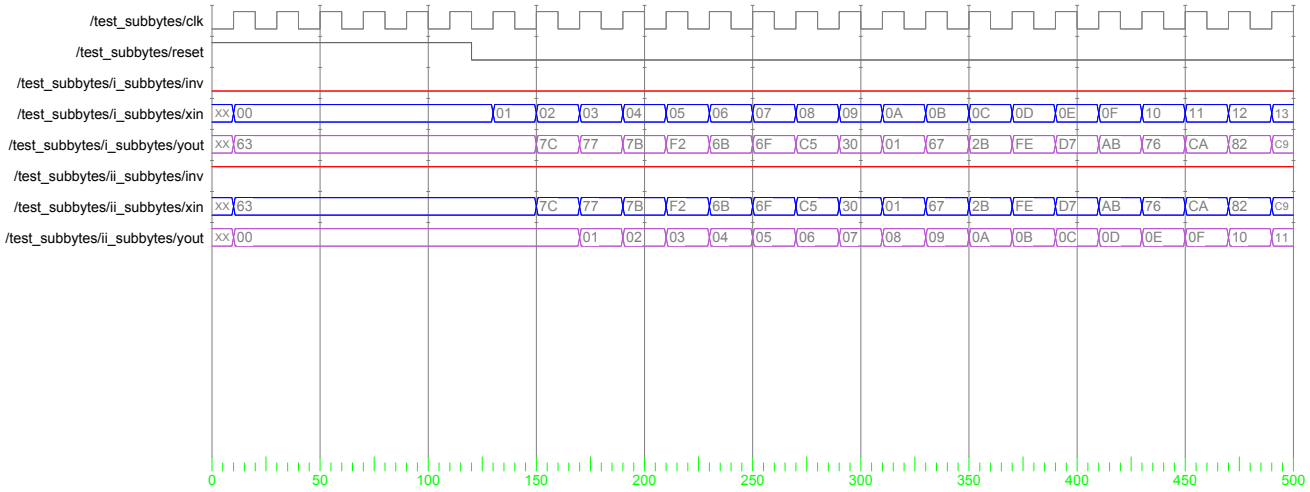
[2] Cillian O’Driscoll, “A Note on Composite Galois Fields and Their Application to Rijndael”, December 2001.

[3] National Institute of Standards and Technology (NIST), “Advanced Encryption Standard,“ *Federal Information Processing Standards Publication 197*, November 2001.

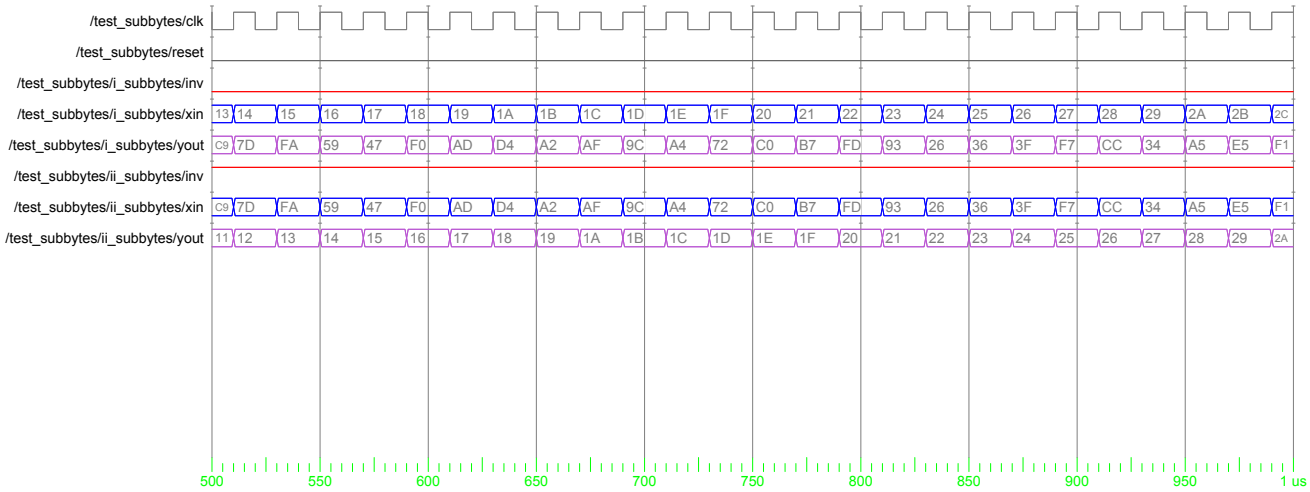
[4] Sumio Morioka and Akashi Satoh, “An Optimized S-Box Circuit Architecture for Low Power AES Design”.

[5] Sumio Morioka and Akashi Satoh, “A 10 Gbps Full-AES Crypto Design with a Twisted-BDD S-Box Architecture,” *Proc. IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD’02)1063-6404/02*, 2002.

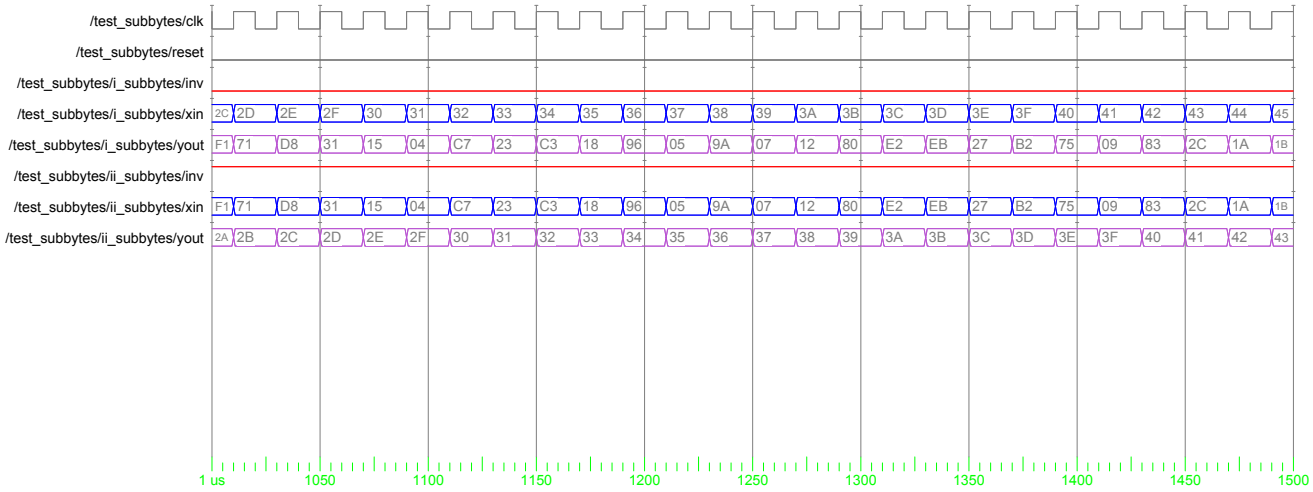
Appendix

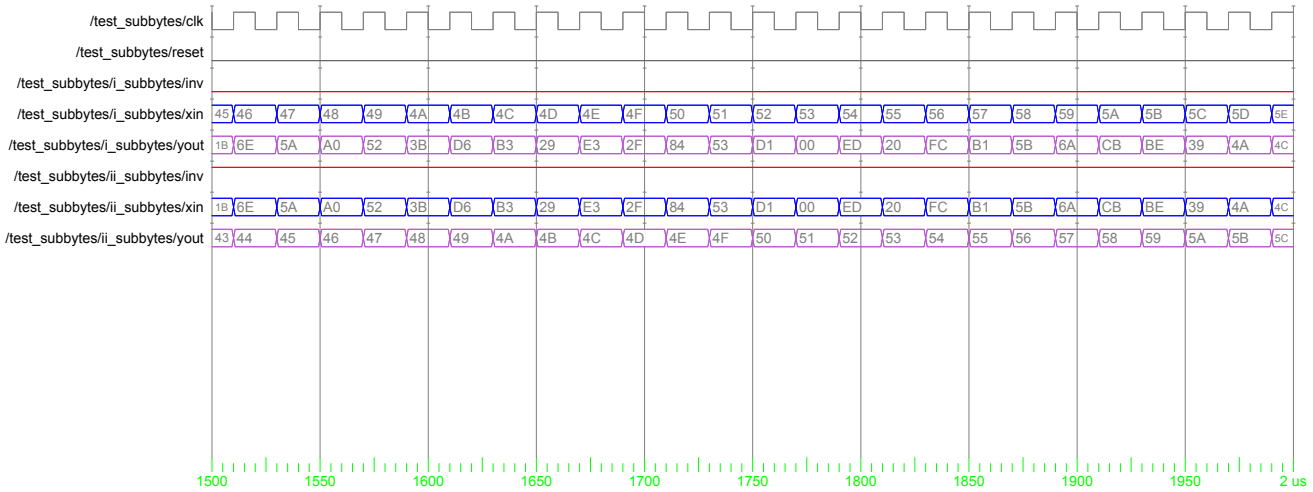


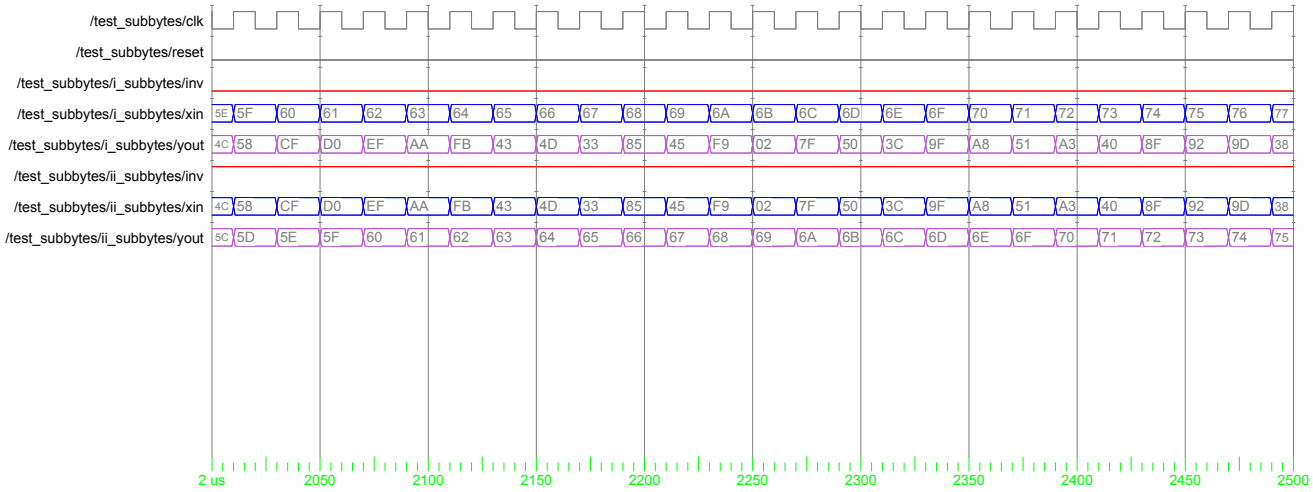
Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 1 Page: 1



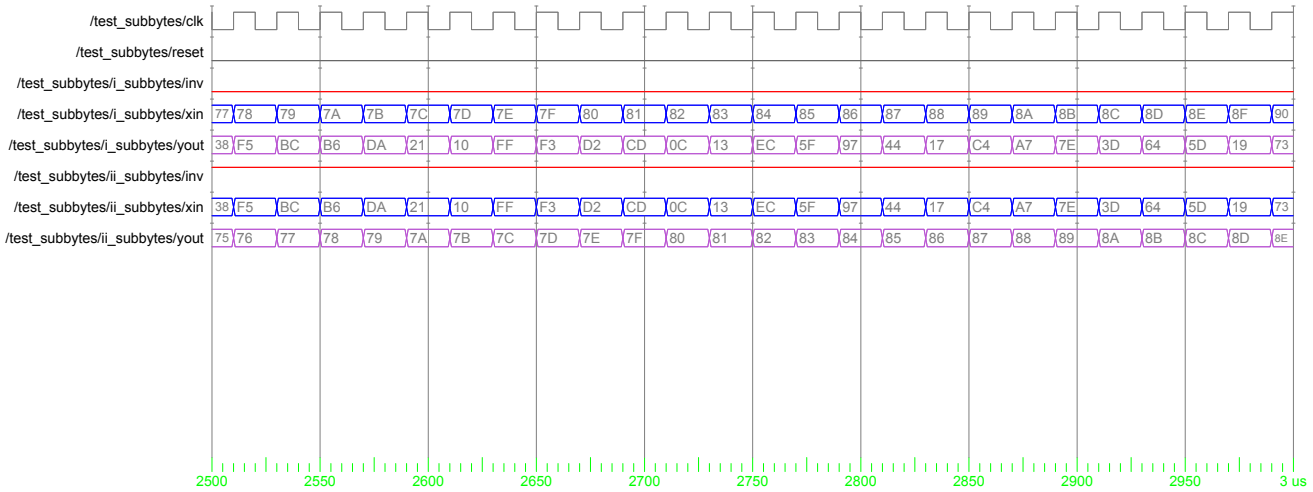
Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 2 Page: 2


/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45
F1 71 D8 31 15 04 C7 23 C3 18 96 05 9A 07 12 80 E2 EB 27 B2 75 09 83 2C 1A 1B
F1 71 D8 31 15 04 C7 23 C3 18 96 05 9A 07 12 80 E2 EB 27 B2 75 09 83 2C 1A 1B
2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43
1 us
1050
1100
1150
1200
1250
1300
1350
1400
1450
1500

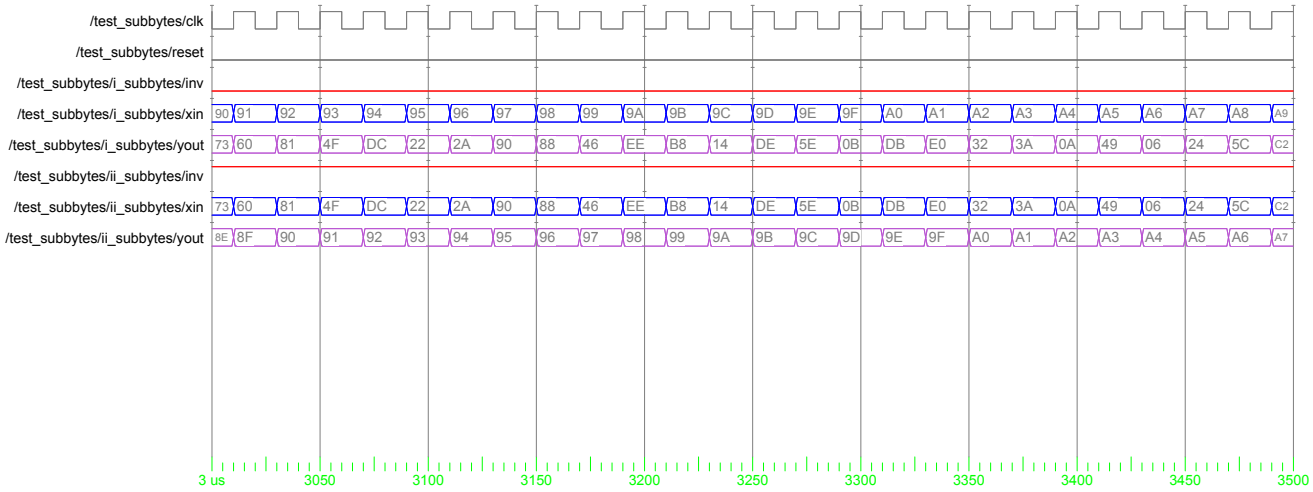

/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E
1B 6E 5A A0 52 3B D6 B3 29 E3 2F 84 53 D1 00 ED 20 FC B1 5B 6A CB BE 39 4A 4C
1B 6E 5A A0 52 3B D6 B3 29 E3 2F 84 53 D1 00 ED 20 FC B1 5B 6A CB BE 39 4A 4C
43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C
1500
1550
1600
1650
1700
1750
1800
1850
1900
1950
2 us

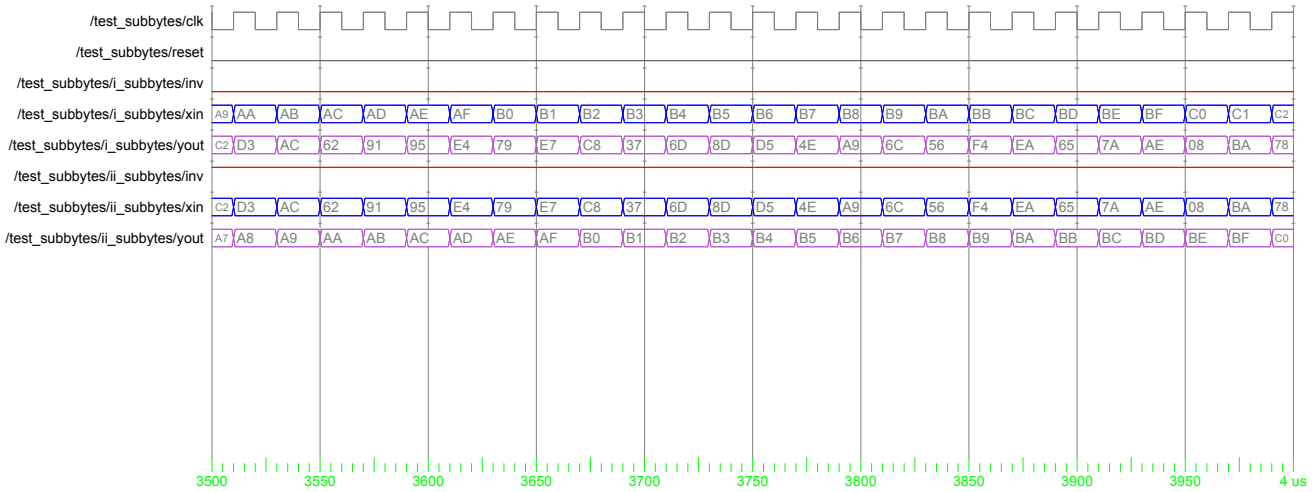


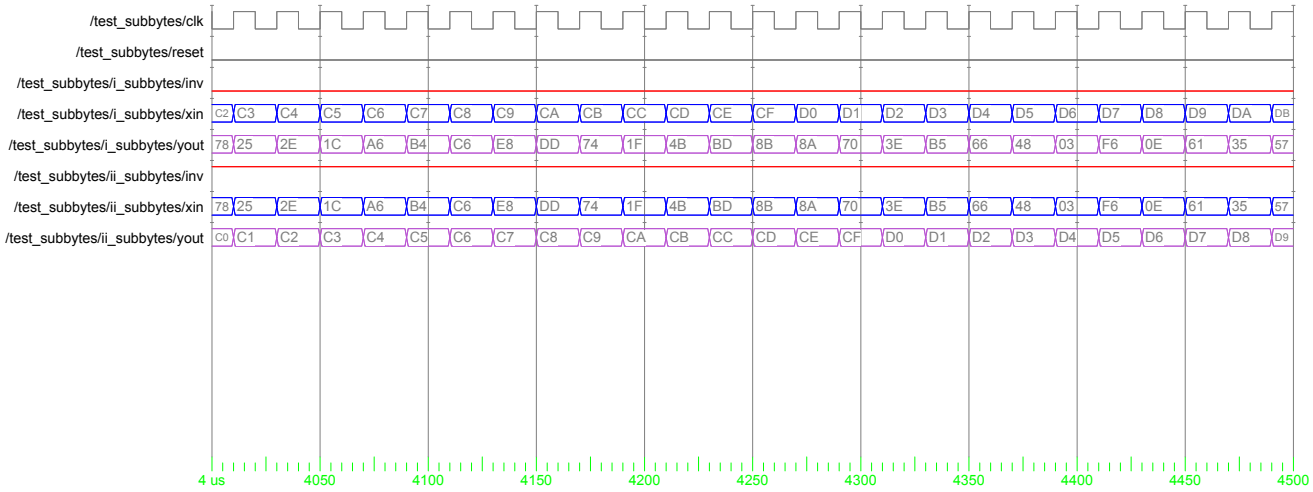
Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 5 Page: 5



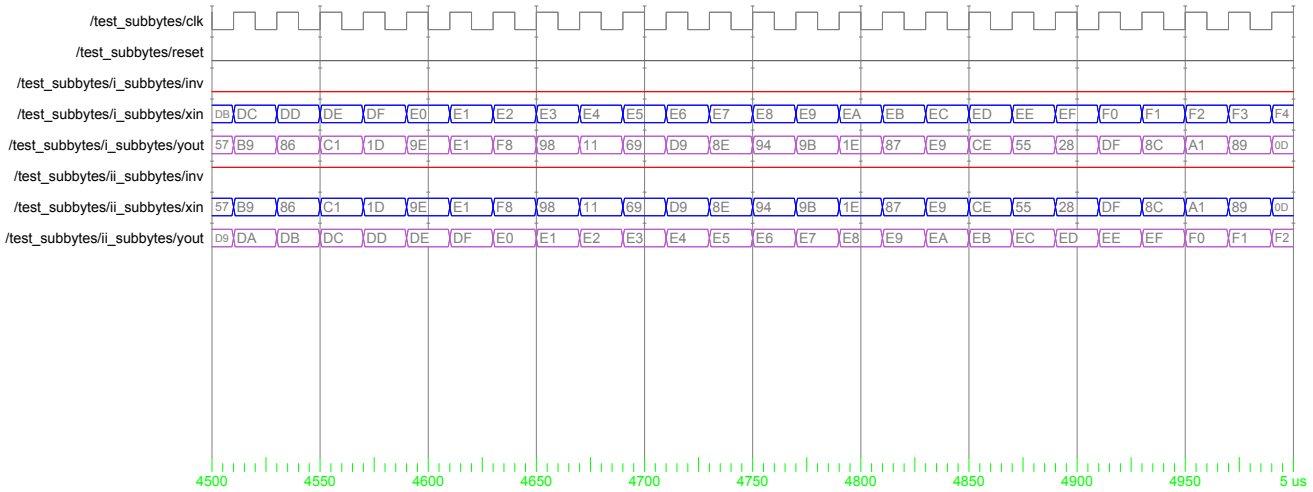
Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 6 Page: 6


/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1 A2 A3 A4 A5 A6 A7 A8 A9
73 60 81 4F DC 22 2A 90 88 46 EE B8 14 DE 5E 0B DB E0 32 3A 0A 49 06 24 5C C2
73 60 81 4F DC 22 2A 90 88 46 EE B8 14 DE 5E 0B DB E0 32 3A 0A 49 06 24 5C C2
8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1 A2 A3 A4 A5 A6 A7
3 us
3050
3100
3150
3200
3250
3300
3350
3400
3450
3500

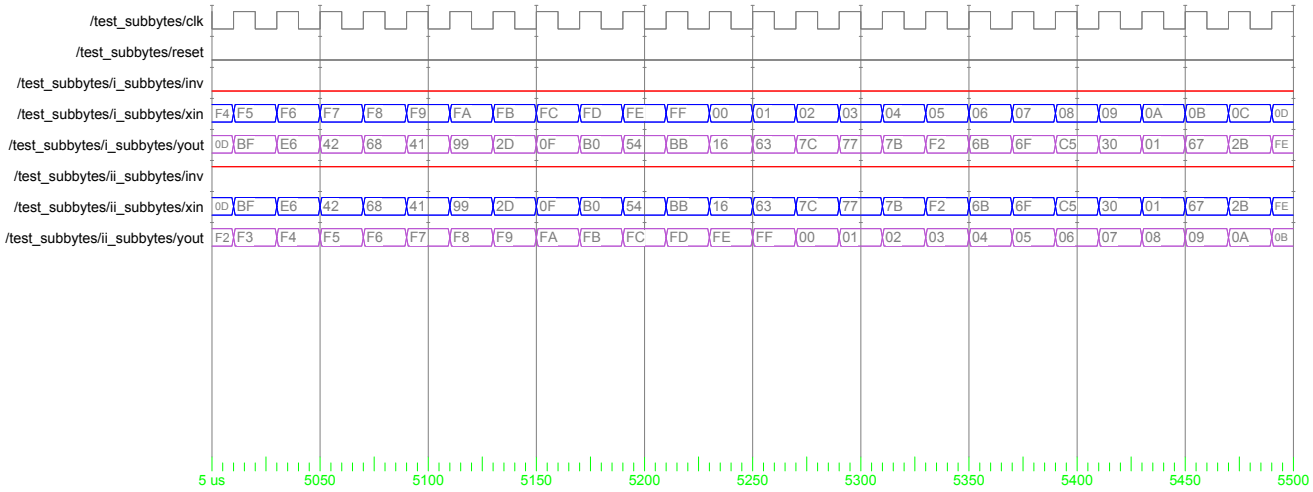

/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF C0 C1 C2
C2 D3 AC 62 91 95 E4 79 E7 C8 37 6D 8D D5 4E A9 6C 56 F4 EA 65 7A AE 08 BA 78
C2 D3 AC 62 91 95 E4 79 E7 C8 37 6D 8D D5 4E A9 6C 56 F4 EA 65 7A AE 08 BA 78
A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF C0
3500
3550
3600
3650
3700
3750
3800
3850
3900
3950
4 us

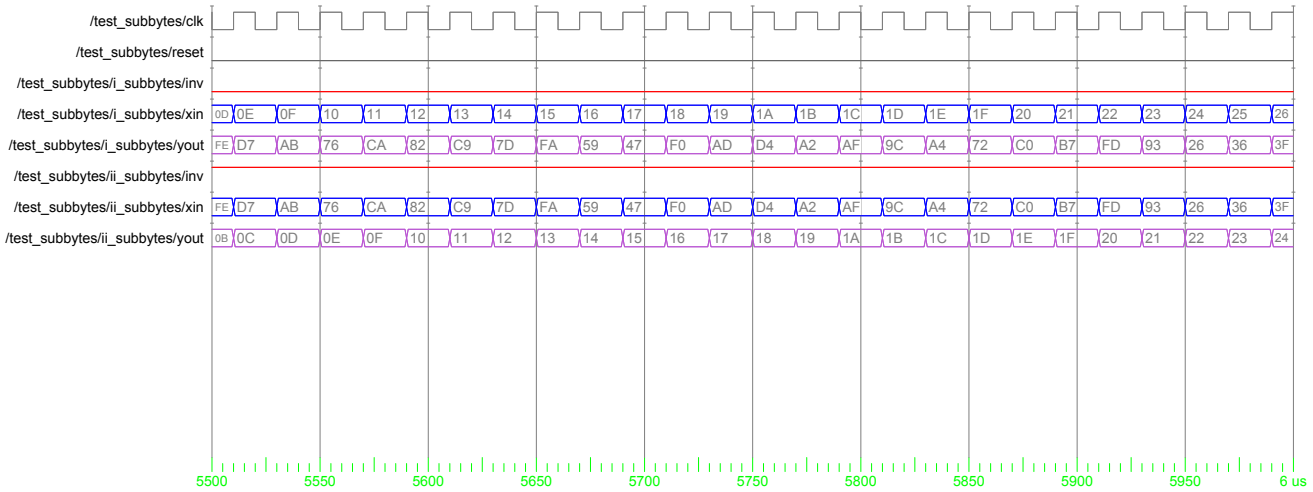


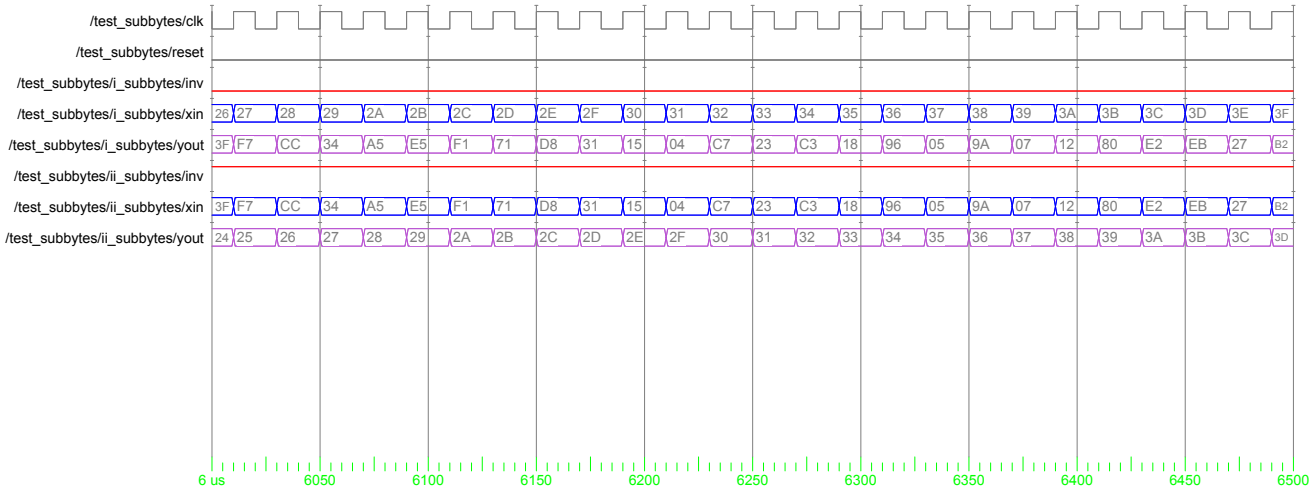
Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 9 Page: 9



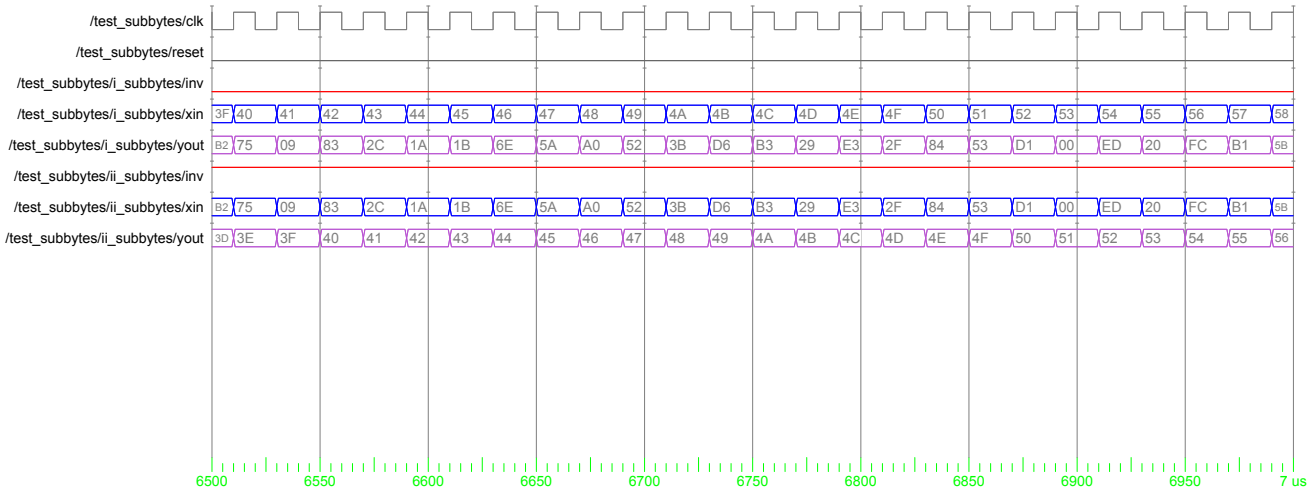
Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 10 Page: 10


/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D
0D BF E6 42 68 41 99 2D 0F B0 54 BB 16 63 7C 77 7B F2 6B 6F C5 30 01 67 2B FE
0D BF E6 42 68 41 99 2D 0F B0 54 BB 16 63 7C 77 7B F2 6B 6F C5 30 01 67 2B FE
F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF 00 01 02 03 04 05 06 07 08 09 0A 0B
5 us
5050
5100
5150
5200
5250
5300
5350
5400
5450
5500

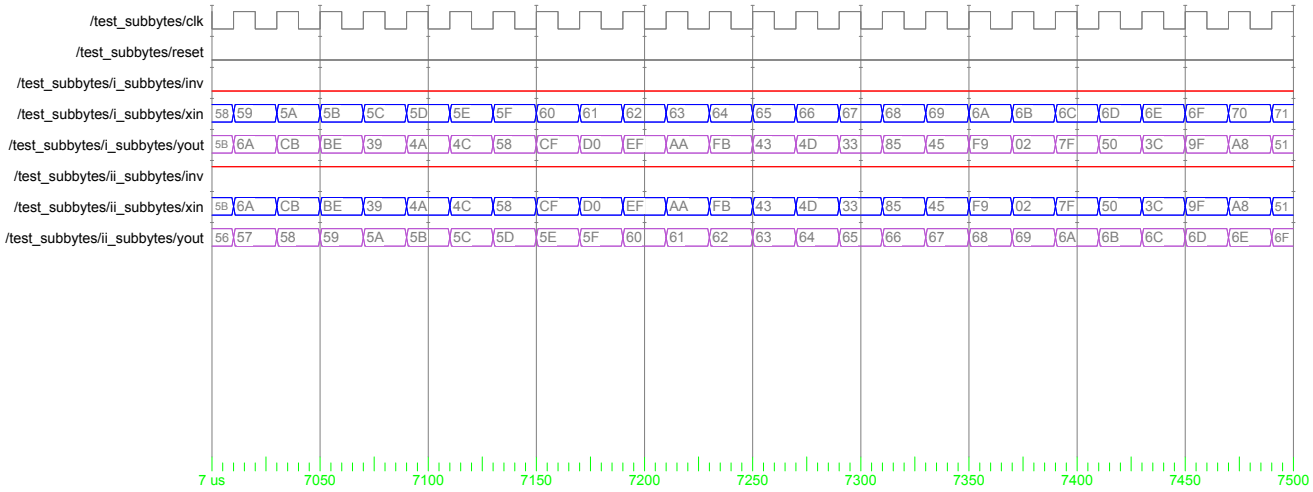

/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26
FE D7 AB 76 CA 82 C9 7D FA 59 47 F0 AD D4 A2 AF 9C A4 72 C0 B7 FD 93 26 36 3F
FE D7 AB 76 CA 82 C9 7D FA 59 47 F0 AD D4 A2 AF 9C A4 72 C0 B7 FD 93 26 36 3F
0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24
5500
5550
5600
5650
5700
5750
5800
5850
5900
5950
6 us



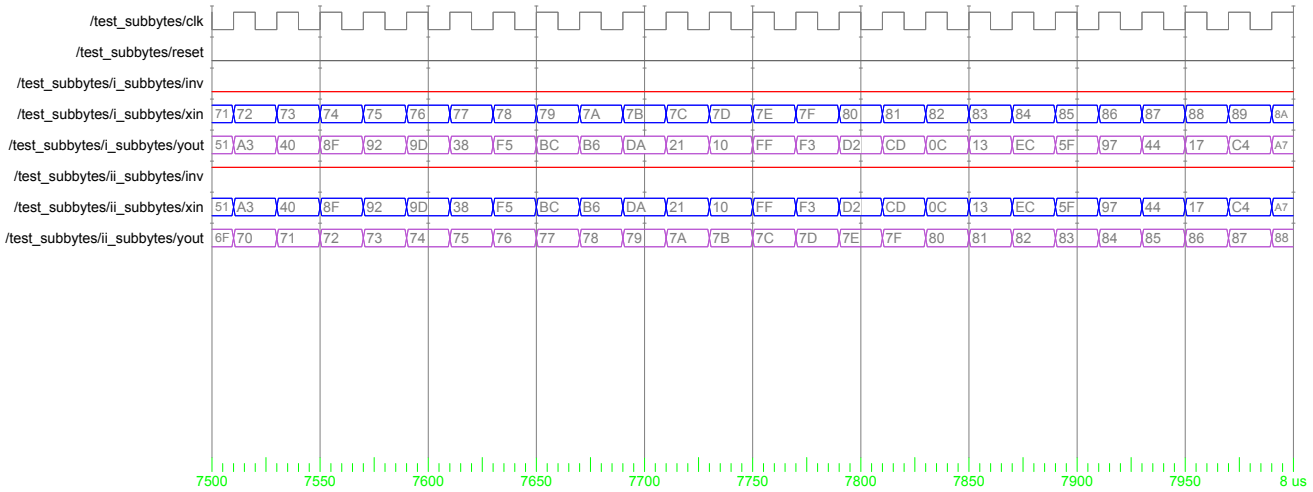
Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 13 Page: 13



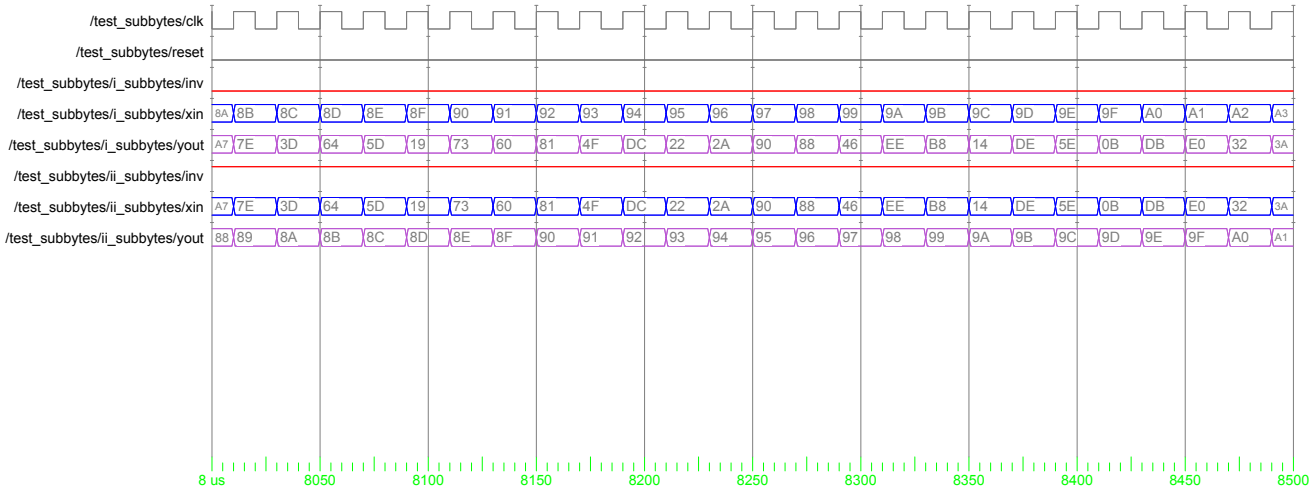
Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 14 Page: 14

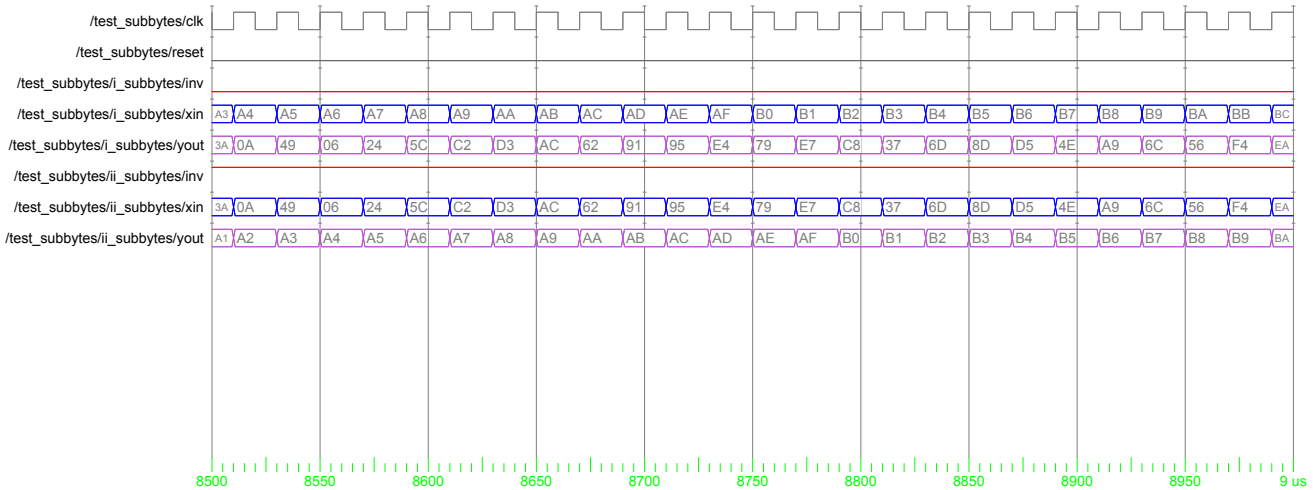


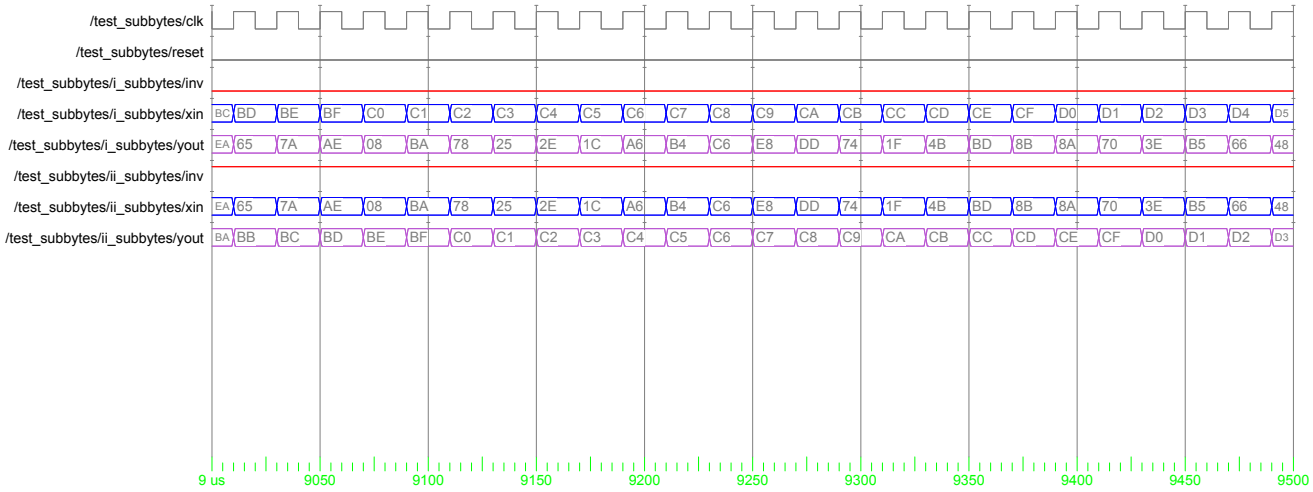
Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 15 Page: 15

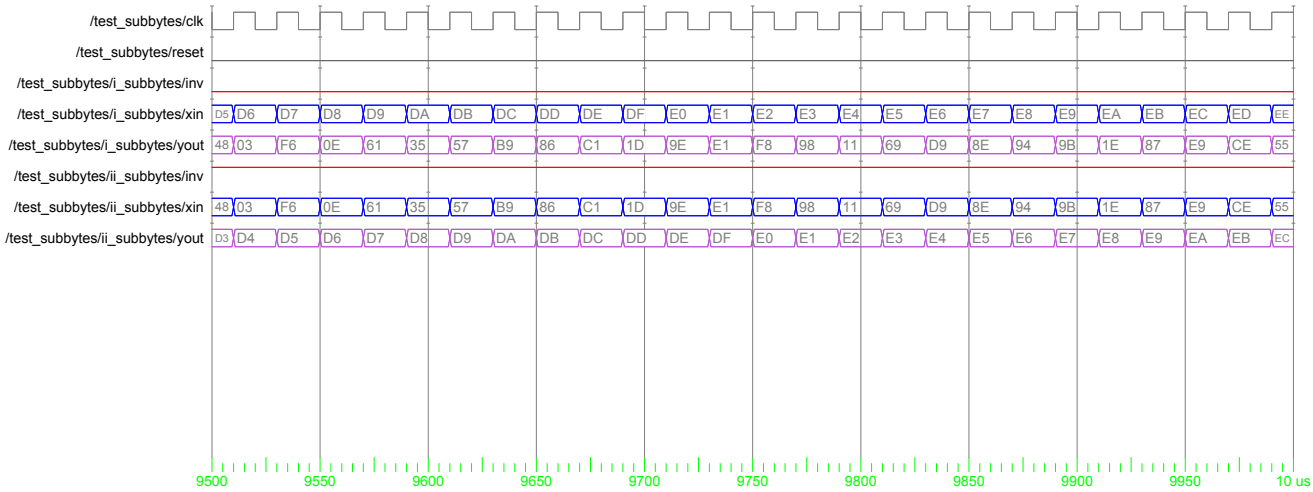


Entity:test_subbytes Architecture:testbench Date: Thu Feb 05 16:45:20 SE Asia Standard Time 2004 Row: 16 Page: 16


/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1 A2 A3
A7 7E 3D 64 5D 19 73 60 81 4F DC 22 2A 90 88 46 EE B8 14 DE 5E 0B DB E0 32 3A
A7 7E 3D 64 5D 19 73 60 81 4F DC 22 2A 90 88 46 EE B8 14 DE 5E 0B DB E0 32 3A
88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1
8 us
8050
8100
8150
8200
8250
8300
8350
8400
8450
8500


/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC
3A 0A 49 06 24 5C C2 D3 AC 62 91 95 E4 79 E7 C8 37 6D 8D D5 4E A9 6C 56 F4 EA
3A 0A 49 06 24 5C C2 D3 AC 62 91 95 E4 79 E7 C8 37 6D 8D D5 4E A9 6C 56 F4 EA
A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA
8500
8550
8600
8650
8700
8750
8800
8850
8900
8950
9 us


/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
BC BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5
EA 65 7A AE 08 BA 78 25 2E 1C A6 B4 C6 E8 DD 74 1F 4B BD 8B 8A 70 3E B5 66 48
EA 65 7A AE 08 BA 78 25 2E 1C A6 B4 C6 E8 DD 74 1F 4B BD 8B 8A 70 3E B5 66 48
BA BB BC BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3
9 us
9050
9100
9150
9200
9250
9300
9350
9400
9450
9500


/test_subbytes/clk
/test_subbytes/reset
/test_subbytes/i_subbytes/inv
/test_subbytes/i_subbytes/xin
/test_subbytes/i_subbytes/yout
/test_subbytes/ii_subbytes/inv
/test_subbytes/ii_subbytes/xin
/test_subbytes/ii_subbytes/yout
D5 D6 D7 D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE
48 03 F6 0E 61 35 57 B9 86 C1 1D 9E E1 F8 98 11 69 D9 8E 94 9B 1E 87 E9 CE 55
48 03 F6 0E 61 35 57 B9 86 C1 1D 9E E1 F8 98 11 69 D9 8E 94 9B 1E 87 E9 CE 55
D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC
9500
9550
9600
9650
9700
9750
9800
9850
9900
9950
10 us