Workshop on Cryptology and Information Security

The Design of Math Core in CPLD for the AES Application

M.H. Jing, Y.H. Chen and C. H. Hsu

Abstract

In contemporary communication system design, the SoC integrates more and more IPs to perform the functions such as error correction code and cryptography. In cryptography, the AES needs much more powerful math modules with high flexibility and operational diversity to cope with the attack from outside [1]. The CPLD/FPGA is usually chosen for the purposes of simple circuit implement, functional design and testing, IP proof and system integration. The advantages of CPLD/FPGA are high efficiency, flexibility and reconfiguration. In the future, more applications will be developed on CPLD/FPGA instead of VLSI. As AES application needs more flexible transformations to design for diversity, the implementation of math IPs in finite field into programmable devices is very important. Without increasing the complexity of IPs, we propose a modified architecture of math modules for CPLD to increase the overall efficiency and keep high-speed performance [2-7]. In those modules, a multiplicative inverse is specially designed for the application on the fast and flexible system.

M.H. Jing and **C.H. Hsu** are with the department of Information Engineering, I-Shou University, Ta-Hsu Hsiang, Kaohsiung, Taiwan, 840. mhjing@isu.edu.tw
TEL: 886-7-6577252 Fax: 6578944
Y.H. Chen is with the Dept. of Electric Engineering, I-Shou University. d880103d@isu.edu.tw

Keywords: Finite Field**,** Multiplier, VLSI, FPGA, MegaCore

The Design of Math Core in CPLD for the AES Application

M.H. Jing, Y.H. Chen and C. H. Hsu

Keywords: Finite Field**,** Multiplier, VLSI, FPGA, MegaCore

Abstract

In contemporary communication system design, the SoC integrates more and more IPs to perform the functions such as error correction code and cryptography. In cryptography, the AES needs much more powerful math modules with high flexibility and operational diversity to cope with the attack from outside [1]. The CPLD/FPGA is usually chosen for the purposes of simple circuit implement, functional design and testing, IP proof and system integration. The advantages of CPLD/FPGA are high efficiency, flexibility and reconfiguration. In the future, more applications will be developed on CPLD/FPGA instead of VLSI. As AES application needs more flexible transformations to design for diversity, the implementation of math IPs in finite field into programmable devices is very important. Without increasing the complexity of IPs, we propose a modified architecture of math modules for CPLD to increase the overall efficiency and keep high-speed performance [2-7]. In those modules, a multiplicative inverse is specially designed for the application on the fast and flexible system.

I. Introduction

In contemporary communication system design, the theory of finite field in modern algebra is widely adopted. The applications of finite field on communication and cryptography include Reed-Solomon codes and Advanced Encryption Standard (AES). In such VLSI system designs, the Ips (Intellectual Properties) are always considered as the critical modules and the highest priority. There are a number of designs of math IPs on finite field such as the multiplier and multiplicative inverse modules. Among those modules, the inverse module is regarded as the key component to affect the overall performance of the system.

From the recent researches, the Look-up tables (LUTs) are the easiest way to implement the various modules in AES. The aim of many papers is to reduce LUTs by using Field Programmable Gate Array (FPGA) [8, 9]. In those papers, the various designs are presented by a huge number of LUTs. However, the implementations of AES by LUTs will increase the IC size and slow down the system.

The advantage of the implementation on FPGA has two folds, such as the reconfigurability and diversity of the system [1]. The design with LUTs may provide off-line reconfiguration but no diversity absolutely. Currently, Jing [2] presented a reordering multiplier that doubles the operation speed in finite-field modules. In this paper, we intend to apply this IP into the format of a gate array (GA) as a standard core to support the finite field computations. Programmable logic encompasses all digital logic circuits configured by end users from low density to FPGAs and complex PLDs (CPLDs)[4]. In order to fit these finite field multipliers into FPGA/CPLD, we do an analysis and propose a new architecture of logic elements. Fast speed by the shorter critical paths and fewer LEs (logic elements) in FPGA becomes an important research topic. Since the major component in inverse module is the multiplier, the design of a fast inverse module in CPLD is feasible. In this paper, a new set of multiplier and inverse modules is proposed as a MegaCore for CPLD on Altera Cooperation.

The Nation Science Council NSC90-2213-E-214-004 in Taiwan supports this work.

M.H. Jing and **C.H. Hsu** are with the department of Information Engineering, I-Shou University, Ta-Hsu Hsiang, Kaohsiung, Taiwan, 840. mhjing@isu.edu.tw
TEL: 886-7-6577252
Y.H. Chen is with the Dept. of Electric Engineering, I-Shou University. d880103d@isu.edu.tw

II. FPGA/CPLD architecture Overview

Programmable logic devices have been presented as a standard, off-the-shelf and configurable integrated circuits (ICs) in the design of SoC (System on a Chip). In early 1980's, the simple GA/PLDs were typically used to integrate multiple discrete logic circuits for experiments and fast prototyping. Today, high-density FPGA/PLDs are employed for the purposes of IP proof, large scale testing support and system-level integration. For small amount of products, they are often preferred as an alternative to application-specific integrated circuits (ASICs) or application-specific standard products (ASSPs).

FPGA/PLDs are offered in different architectures. And a variety of memory elements are available for device configuration. Moreover, CPLDs and FPGAs have different interconnect structures. The segmented interconnect structure of FPGAs uses multiple metal lines of varying lengths, joined by pass transistors, anti-fuses and switching matrices. Additionally, the interconnect structures eliminate the unpredictable timing associated with a segmented interconnect structure and provide fast and fixed delay paths between logic cells by offering predictable timing [3]. The structure of FPGA/CPLD makes the SoC design easier in various development stages.

III. The CPLD/FPGA Families

The APEX 20K CPLD of Altera Cooperation has a relatively fine-grained architecture with a low delay hierarchical routing scheme. The APEX 20K device incorporates LUT-based logic, product-term-based logic and memory into one device. The ESB can implement a variety of memory functions, including CAM, RAM, dual-port RAM, ROM, and FIFO functions. In the functional blocks, the APEX 20K devices are constructed from a series of MegaLAB structures. Each MegaLAB structure contains 16 logic array blocks (LABs), one ESB, and a MegaLAB interconnect, which routes signals within the MegaLAB structure.

Each LAB consists of 10 logic elements (LEs). The LEs are associated with carry and cascade chains, LAB control signals, and the local interconnect. The LEs are the smallest unit of logic in the APEX 20K architecture. Each LE contains a four-input LUT, as a function generator that can be quickly implemented as every function with four variables and one output. In addition, each LE contains a programmable register, carry and cascade chains, as shown in Figure 1 [3].

An FPGA from Xilinx Cooperation consists of an array of functional blocks along with interconnection channels and matrices. The functional blocks can be programmed to implement combinational or sequential logic functions. Several FPGA architectures have been developed for different applications. The most widely used logic block is the 5-input look-up table (LUT) based on FPGA. Each logic block in FPGA has dual 4-input LUT and is integrated by a 3-input LUT. However, the dual 4-input LUT architecture of Xilinx cannot be reconfigured into two separate LUTs, and it is not suitable for finite field applications. So, we prefer the Altera CPLD with four-input LUT architecture.

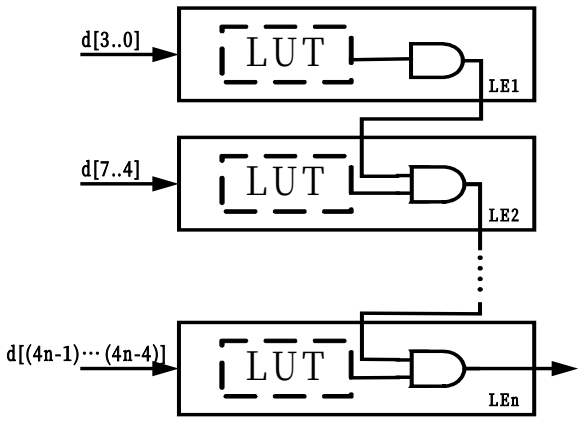

Figure 1. The AND Cascade Chain in LAB

From Jing [2], a high-speed parallel multiplier in finite-field is developed and named the reordering multiplier. Two math modules are mapped into logic blocks in CPLD and will be presented in following sections.

IV. The basic math in $GF(2^m)$

For m a positive integer, let $GF(2^m)$ be the finite field containing 2^m -1 elements. Then, $GF(2^m)$ is a vector space over $GF(2)$ of dimension m. There is a basis of the form $\{1, \alpha, \Lambda, \alpha^{m-1}\}$ with $\alpha \in GF(2^m)$. Elements in $GF(2^m)$ can be viewed as polynomials with coefficients in $GF(2)$, namely, $A(\alpha) = a_{m-1}\alpha^{m-1} + a_{m-2}\alpha^{m-2} + \Lambda + a_1\alpha + a_0$ where $a_0, a_1, \Lambda, a_{m-1} \in GF(2)$. If $F(x)$ is the minimal polynomial of α over $GF(2)$, the polynomial $F(\alpha)$, $x^m + f_{m-1}x^{m-1} + ... + f_0$, is used as the generator polynonial.

Let $A(\alpha) = \sum_{i=0}^{m-1} a_i \alpha^i$ and $B(\alpha) = \sum_{i=0}^{m-1} b_i \alpha^i$ be two elements in $GF(2^m)$. If $C(\alpha) = \sum_{i=0}^{m-1} c_i \alpha^i \in GF(2^m)$ is the product of $A(\alpha)$ and $B(\alpha)$, then

$$C(\alpha) \equiv A(\alpha) \cdot B(\alpha) \mod F(\alpha). \tag{1}$$

Denoting $\mathrm{A} = A(\alpha)$ and expanding the right-hand side of Eq. (1), one has

$$C(\alpha) \equiv (\mathrm{A} \cdot b_{m-1}\alpha^{m-1} + \mathrm{A} \cdot b_{m-2}\alpha^{m-2} + \Lambda + \mathrm{A} \cdot b_0\alpha^0) \mod F(\alpha). \tag{2}$$

From Jing [2], the recursive in Eq. (2) can be used to construct a parallel establishment and present the multiplier. For the multiplier in AES, one uses the polynomial $F(x) = x^8 + x^4 + x^3 + x + 1$ as an example to illustrate the design of multiplier. The multiplier can be partitioned into two groups: the regular group and the reordering one, as shown in Figure 2. According to [1], the vector $\boldsymbol{d} = \{d_0, d_1, \mathrm{K}, d_6\}$ can be calculated as follow:

$$\begin{aligned}
d_0 &= a_7 b_7 \\
d_1 &= \sum_{i=0}^{1} a_{7-i} b_{6+i} \\
d_2 &= \sum_{i=0}^{2} a_{7-i} b_{5+i} \\
d_3 &= \sum_{i=0}^{3} a_{7-i} b_{4+i} \\
d_4 &= \sum_{i=0}^{4} a_{7-i} b_{3+i} + d_0 \\
d_5 &= \sum_{i=0}^{5} a_{7-i} b_{2+i} + d_0 + d_1 \\
d_6 &= \sum_{i=0}^{6} a_{7-i} b_{1+i} + d_2 + d_3.
\end{aligned} \tag{3}$$

Then, the result $C(\alpha) = \{c_0, c_1, \mathrm{K}, c_7\}$ is

$$\begin{aligned}
c_0 &= \sum_{i=0}^{6} a_{7-i} b_{1+i} + a_0 b_0 + d_1 + d_2 \\
c_1 &= \sum_{i=0}^{5} a_{7-i} b_{2+i} + \sum_{i=0}^{1} a_{1-i} b_i + d_0 + d_1 + d_6 \\
c_2 &= \sum_{i=0}^{4} a_{7-i} b_{3+i} + \sum_{i=0}^{2} a_{2-i} b_i + d_0 + d_5 \\
c_3 &= \sum_{i=0}^{3} a_{7-i} b_{4+i} + \sum_{i=0}^{3} a_{3-i} b_i + a_0 b_3 + d_4 + d_6 \\
c_4 &= \sum_{i=0}^{2} a_{7-i} b_{5+i} + \sum_{i=0}^{4} a_{4-i} b_i + d_3 + d_5 + d_6 \\
c_5 &= \sum_{i=0}^{1} a_{7-i} b_{6+i} + \sum_{i=0}^{5} a_{5-i} b_i + d_2 + d_4 + d_5 \\
c_6 &= a_7 b_7 + \sum_{i=0}^{6} a_{6-i} b_i + d_1 + d_3 + d_4 \\
c_7 &= \sum_{i=0}^{7} a_{7-i} b_i + d_0 + d_2 + d_3.
\end{aligned} \tag{4}$$

V. Mapping the multiplier into CPLD

The specified multiplier can be implemented with Eqs. (3) and (4). With the presentation of the repositioned d_x, the overall circuit diagram is shown in Figure 3 [4]. Here, we take the c_3, analyzed as the critical path of this multiplier, to present the proposal design. The signal c_3 is extracted and shown in Figure 4.

From Figure 4, AND array and XOR arrays are the main components of a multiplier. To map into CPLD, the AND array is firstly implemented by the product-term-based logic. Although the XOR array may be implemented by those 4-input LUTs, It is best to add an XOR in parallel with the AND/OR gate in the cascade chain from Figure 1. The reordering multiplier may be used as the new cascade chain function to reduce the number of LEs. For example, the signal c_3 from Eq. (4) is shown in Figure 5, in which only 3 LEs is used. The complete 8 bit-reordering multiplier is just under one LAB, shown in Figure 6.

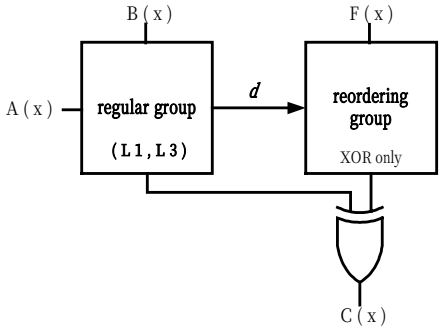

Figure 2. The block diagram of reordering multiplier.

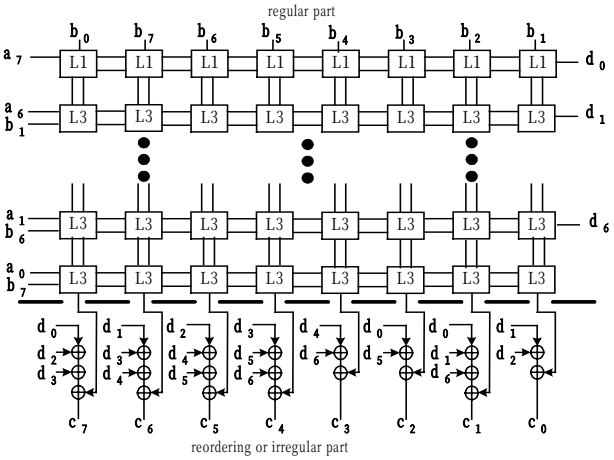

Figure 3. A multiplier with repositioned XOR gates

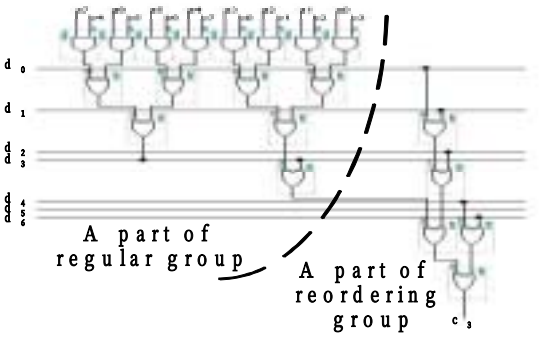

Figure 4. The implementation of c_3.

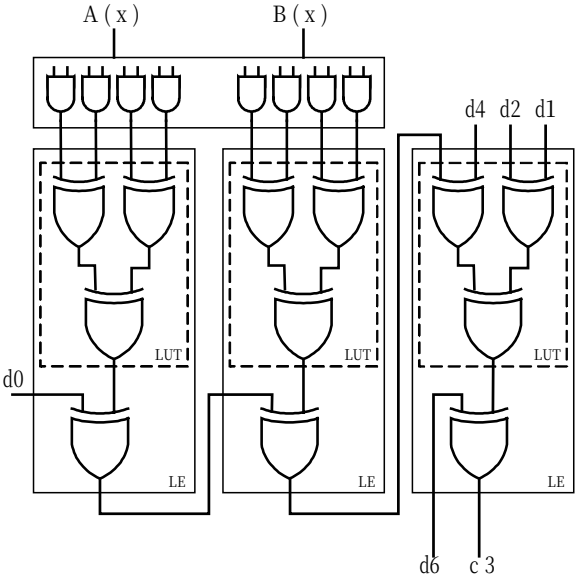

Figure 5. The c_3 generated from the proposed LEs.

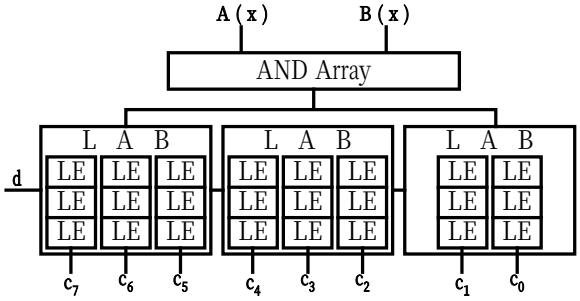

Figure 6. The architecture of the 8-bit multiplier

VI. Multiplicative inverse in $GF(2^m)$

In finite field $GF(2^m)$, the AES has m=8. The multiplicative inverse A^{-1} is $A^{2^m-2} = A^{254}$. Basically, A^{254} can be purely organized by multiple multipliers and A^2 modules as shown in Figure 7 [5]. But its performance is very poor. So, the power circuit, A^{2^i}, are used to speed up A^{-1} with smaller circuit in size [5,6]. From our previous study, a more uniform structure of A^{-1} is shown in Figure 8 [7]. In finite field, when $A = \sum_{i=0}^{m-1} a_i \alpha^i$, we get

$$A^2 = (\sum_{i=0}^{m-1} a_i \alpha^i)^2 \bmod F(\alpha),$$

the $A^2 = A \otimes A \bmod F(\alpha) = A^{2^1}$, $A^4 = A^{2^2}$. So the poser module, A^{2^i}, is not complicated. Tables 1 and 2 have shown the coefficients of A^2 and A^4. Each A^{2^i} module in a 8-bit application can be implemented by a single LAB (or less than 10 LEs). In CPLD, each megaLAB has 16 LABs for the design of a set of major function, enough for multiple multipliers and power modules. And the multiplicative inverse into a single MegaLAB is shown in Figure 9.

Inside the LE, there is macro-cell to configure the module into various structures such as combinational or sequential circuit that may configure the system into parallel or pipelined architecture. The multiplicative inverse fits into a single MegaLAB with either parallel or pipelined architecture. This means that the system may switch to a pipelined design for N-time faster clock without extra cost. The critical path of the pipelined architecture is the multiplier; however, the delay of this path is much faster than the LUT. The benefit is that the clock of a pipelined architecture is faster than the implementation by using the LUTs. According to the simulation, the clock speed is two times faster.

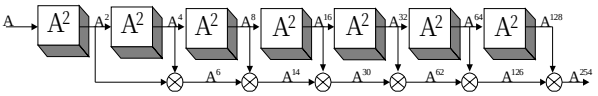

Figure 7. The A^{-1} module using seven A^2

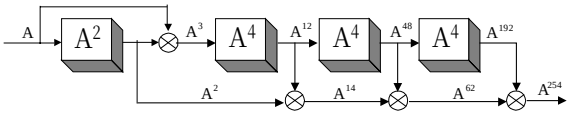

Figure 8. The modified multiplicative inverse module

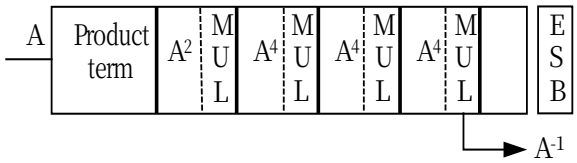

Figure 9. The multiplicative inverse module in CPLD

Table 1. The coefficients of **C**= A^2

c_7	$a_7 + a_6$
c_6	$a_5 + a_3$
c_5	$a_6 + a_5$
c_4	$a_7 + a_4 + a_2$
c_3	$a_7 + a_6 + a_5 + a_4$
c_2	$a_5 + a_1$
c_1	$a_7 + a_6 + a_4$
c_0	$a_6 + a_4 + a_0$

Table 2. The coefficients of **C**=A^4

c_7	$a_7 + a_6 + a_5 + a_3$
c_6	$a_7 + a_4$
c_5	$a_6 + a_3$
c_4	$a_6 + a_5 + a_4 + a_2 + a_1$
c_3	$a_4 + a_3 + a_2$
c_2	$a_7 + a_5 + a_4$
c_1	$a_6 + a_5 + a_4 + a_3 + a_2$
c_0	$a_7 + a_6 + a_5 + a_3 + a_2 + a_0$

VII. Conclusion

Actually, a FPGA version of AES IP is built in our lab to support our researches. The prototyping system and testing support are using PCI interface on IBM compatible PC. The Windows interface is used to support the experiment data collection and demonstration.

Using a modified LE architecture, the finite-field math modules are implemented into MegaCore and presented as the Library of Parameterized Modules (LPM). Taking advantage of the MegaCore in each LE, the structure of the math core can configured as parallel or pipelined architecture. In this paper, the math modules are easily implemented on CPLD with simple configuration, higher speed and higher performance than the LUT configuration. Taking AES as an example, the speed of this application on CPLD may execute on 80MHz or higher, and the reconfigure ability will increase the diversity on functionality [1]. So, the proposed CPLD architecture can be widely used in different area, such as the Reed-Solomon Code and AES.

VIII. References

[1] M. H. Jing, Y.T. Chang, Y. H. Chen and C.H. Hsu "The Diversity Study for AES on the FPGA Application", The IEEE 1st int. Conf. on Field-Programmable Technology (FPT), Hong Kong, Dec. 16-18, 2002 (submitted)

[2] M. H. Jing, T. K. Truong, Fellow, IEEE, Y.T Chang, Y.H. Chen, C.H. Hsu, and H.C. Wu, "The Design of Low Latency Multiplier for AES ", IKE'02 International Conf., LasVegas, USA, June, 24-27 2002, pp198-203.

[3] Altera Corporation, "APEX 20K Programmable Logic Device Family", Data Sheet, Nov. 1999, Ver. 2.05.

[4] C.H. Hsu, T.K. Truong, Fellow, IEEE, M.H. Jing, W.C. Wu and H.C. Wu, "Feasibility study for the design of a FPGA multiplier-core on Finite Field**",** The IEEE int. Conf. on Field-Prog. Technology (FPT), Hong Kong, Dec. 16-18, 2002 (submitted)

[5] M. H. Jing, Y.T. Chang, Y. H. Chen and C.H. Hsu "The Design of A Fast Inverse Module in AES", International Conference on Info-tech & Info-net, Oct 2001, pp. 298–303.

[6] Koc C. K., Sunar B., **"**Low-complexity bit-parallel canonical and normal basis multipliers for a class of finite fields**",** Computers, IEEE Transactions, Vol. 47 Issue 3, Mar. 1998, pp. 353 –356.

[7] C.H. Hsu, "The Implementation of an AES IP and its Security Analysis". MS Thesis, Department of Information Engineer I-Shou University, July, 2002.

[8] McLoone, W. and McCanny, J.V., "Rijndael FPGA implementation utilizing look-up tables", Signal Processing Systems, 2001 IEEE Workshop on, 2001, pp.349-360

[9] Fiskiran, A.M. and Lee, R.B., "Performance impact of addressing modes on encryption algorithms", Computer Design, 2001. ICCD 2001. Proceedings. 2001 International Conference on, 2001 pp. 542-545